National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

UNIVERSITY OF ALBERTA

Expert Tuning of Multivariable Model-Based Controllers

BY

Simarjit Singh Dhaliwal

©

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH
IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR
THE DEGREE OF
Master of Science
IN
Process Control

Department of Chemical Engineering

EDMONTON, ALBERTA
Fall, 1992

National Library of Canada
Canadian Theses Service
Ottawa, Canada
K1A 0N4

Bibliothèque nationale du Canada
Service des thèses canadiennes

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-77152-6

Canada

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR:	Simarjit Singh Dhaliwal
TITLE OF THESIS:	Expert Tuning of Multivariable Model-Based Controllers
DEGREE:	Master of Science
YEAR THIS DEGREE IS GRANTED:	1992

Permission is hereby granted to UNIVERSITY OF ALBERTA LIBRARY to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

(SIGNED)

PERMANENT ADDRESS:
#434, Sector 6,
Panchkula, Haryana
India

DATED: 8th oct. 1992

UNIVERSITY OF ALBERTA
FACULTY OF GRADUATE STUDIED AND RESEARCH

The undersigned certify that they have read , and recommended to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled *Expert Tuning of Multivariable Model-Based Controllers* submitted by *Simarjit Singh Dhaliwal* in partial fulfilment of the degree of *Master of Science* in *Process Control.*

Prof. S.L.Shah (supervisor)

Prof. D.G.Fisher

Prof. K.Bollinger

Dated: 8th oct. 1992

to my parents and Rema

Abstract

Industry faces a continuing challenge in operating their plants economically and safely. Among the many day to day plant operations required to meet these objectives are the challenges of initial tuning of controllers and always keeping them in "tune", plus supervising and monitoring the process. The job of tuning controllers is especially tedious for Long Range Predictive Control (LRPC) type strategies such as DMC (Cutler and Ramaker, 1980) and GPC (Clarke *et al.*,1987), since there are a relatively large number of design and operating parameters that can be adjusted and there is a lack of understanding and knowledge of these algorithms by most plant personnel. This thesis focusses on the use of an expert system to tune a (non-adaptive) GPC algorithm for industrial process applications.

An experimental evaluation of an expert systems based tuning strategy for LRPC algorithms is performed. To the best knowledge of the author, there is no such reported experimental evaluation in the literature. The experiments are performed on a 2-input 2-output highly interactive process. Experimental results indicate that closed loop system performance is enhanced when the tuning strategy is active. Also, irrespective of the values of initial tuning parameters, this strategy converges on tuning parameters which result in a stable closed loop with good tracking properties.

The Adaptive Long range predictive control Performance Supervisor or ALPS (Walther, 1992) is used to tune the parameters of the Generalized Predictive Control (GPC) algorithm when applied to a 5-input, 5-output heavy oil fractionater (Shell bench mark Problem). Gensym's G2, a real-time, object-oriented, commercial expert systems package was used for developing ALPS. The tuning strategy is found to work well in improving the closed loop performance of the heavy oil fractionater.

Experience gained from simulations of the heavy oil fractionater has been translated into rules which are applicable to most multi-input multi-output (MIMO) systems. These rules are used to choose *initial* tuning parameters of MIMO GPC.

Certain rules, such as those used to calculate the rise time of an output had

to be modified to handle the larger 5-input 5-output application. The exhaustive approach used by Walther (1992) in developing rules for calculating the rise time for a 2-input 2-output system results in 2 rules, but for a 5-input 5-output system using the same approach will result in 120 rules. This is clearly very tedious and inefficient. In order to avoid this problem, a non-exhaustive approach is taken to write these rules, by using a few procedures which simplify the logic.

Simulations of the heavy oil fractionater with model plant mismatch show that increasing the input weighting, λ, in proportion to the sensitivity of the outputs to inputs can eliminate the 'noisy' interaction in the outputs while maintaining good, though relatively slow tracking of the set points. Slow tracking is attributed to a relatively large value of the input weighting.

Another concern of industry is the idea of appropriate use of extra degrees of freedom where available. From a practical perspective, this means that if there is a cost associated with each input, then the total cost of inputs can be 'optimized' by judicious choice of inputs. An initial investigation of this issue is conducted by simulating a 5-input 4-output process. It was observed that the steady state values of the inputs when all of them are used is different from when only 4 of the 5 inputs are used. The existing ALPS knowledge base is enhanced to implement a strategy which uses the extra degree of freedom only when required.

Acknowledgement

A number of people extended their intellectual and emotional support for which I am very grateful. In particular, I would like to thank my parents for showing me the way and believing in education, my wife Rema, without whose love and support none of this would have been possible, and my uncle Mr. H. Dhaliwal who lent me his everlasting support in every sense of the word.

I offer my sincere thanks to Dr. S.L. Shah and Dr. D.G. Fisher for their guidance, patience and the invaluable learning experience which I shared with them. I am grateful to Bob Barton, Paul Barrow and Henry Sit in the DACS centre for helping me out when there was 'computer trouble'. Many thanks are due to Walter Boddez in the instrument shop, Keith Faulder, Ron Van Den Heuvel and Robert Scott in the Machine shop, for their time and patience in fixing the experimental equipment.

The friends that I made in the department were invaluable. My 'office mate' Sreekanth Lalgudi and 'office neighbours' Christine Ho, Kunyu Kwok and Hong Zhou were a lively bunch and made my stay in the department a pleasant one. Some other fellow students whom I will always remember are Wellington Kwok, Jean Corbin, Paul Nawrocki, Don Ryland, Pranob Banerjee, Mary Bourke, Kevin Dorma, Mike Macharia and Adam Rankin.

Special thanks are due to Cindy Heisler for organizing social events within the department, making it a happier place. I also acknowledge the help extended to me by Bev Walker in the department office.

The financial support from NSERC and the Department of Chemical engineering is gratefully acknowledged.

Table of Contents

1 Introduction 1

1.1 Issues in multivariable control 2

1.2 Expert GPC 3

1.3 Objectives of study 5

1.4 Structure of thesis 5

2 Expert Generalised Predictive Control 7

2.1 Long Range Predictive Control 7

2.2 The Multivariable GPC algorithm 10

2.2.1 The process model 10

2.2.2 The objective function 11

2.2.3 Long range prediction of process outputs 13

2.2.4 The multivariable GPC control law 14

2.2.5 Extensions to the multivariable GPC algorithm 15

2.2.5.1 Use of the T-filter to enhance robustness 16

2.2.5.2 Feedforward compensation 16

2.2.5.3 Input and output constraints 17

2.3 Related work 19

2.4 Expert GPC 20

2.4.1 Justification for developing Expert GPC 20

2.4.2 The architecture of expert GPC 21

2.4.3 Choice of hardware and software 23

2.4.4 Tuning rules 25

2.4.4.1 Organization of tuning rules 27

2.4.4.2 Tuning strategy 31

2.4.4.3 Modifications to the knowledge base 36

2.5 Summary 38

3 Issues in multivariable control 40

3.1 Review of previous work on multivariable control 40
3.2 Structure and complexity of multivariable systems 43
3.3 Stability of multivariable systems 45
  3.3.1 Multiple time scale systems 45
  3.3.2 Poorly conditioned systems 47
  3.3.3 The effect of numerical precision on simulation results 57
3.4 Conclusions 62

4 Simulation of a Multivariable System 64

4.1 The Shell Control Problem 64
  4.1.1 Process description 64
  4.1.2 Process model 65
  4.1.3 Control objectives and constraints 68
  4.1.4 Open loop simulation 69
4.2 Control Strategy 71
  4.2.1 Selection of initial tuning parameters 71
  4.2.2 Closed loop simulations 72
4.3 Autotuning GPC on the Shell Problem 75
  4.3.1 Output weighting tuning 76
  4.3.2 T-Filter tuning 79
  4.3.3 Output constraints 82
  4.3.4 Closed loop performance with model plant mismatch 89
  4.3.5 Simulation of a non-square system 95
4.4 Conclusions 103

5 Experimental evaluation of autotuning GPC 106

5.1 Process description 107
  5.1.1 The Continuous Stirred Tank Heater 107
  5.1.2 The OPTO-22 process interface 109
5.2 Process model 112

5.2.1 Step response model 112
5.3 Process interface software development 114
5.3.1 Communication protocol 114
5.3.2 Data integrity and program integration 116
5.4 Experimental results 117
5.4.1 Results with autotuner disabled 119
5.4.2 Results with autotuner enabled 120
5.4.2.1 Results with regulatory supervision 120
5.4.2.2 Results with servo supervision 122
5.4.2.3 Results with regulatory and servo supervision 122
5.5 Conclusions 125

6 Conclusions 129
6.1 Suggestions for future work 133

References 135
Appendix A 141
Appendix B 150
Appendix C 153

List of Tables

2.1 A summary of the source code executed 25

2.2 Correlation among Δt_r and $\Delta N2$ 36

4.1 Transfer functions of the Shell Problem 67

4.2 Uncertainties in the gains, K, given in table 4.1 68

List of Figures

1.1 Block diagram of a 3-input 3-output MIMO system with three SISO feedback controllers 4

2.1 The concept of long range prediction 9
2.2 Block diagram representation of LRPC 9
2.3 Possible functions of the process supervisor 22
2.4 Hierarchy of the expert systems supervisor 24
2.5 The windowed environment of G2 26
2.6 Organization of tuning rules 28

3.1 Simulation of MIMO system (3.7) with sampling time of 2 seconds 48
3.2 Simulation of MIMO system (3.7) with sampling time of 2 seconds 49
3.3 Simulation of MIMO system (3.7) with sampling time of 5 seconds 50
3.4 Simulation of MIMO system (3.7) with sampling time of 5 seconds 51
3.5 Simulation of a poorly conditioned system with λ weighting set to 0.0 53
3.6 Simulation of a poorly conditioned system with λ weighting set to 0.0 54
3.7 Simulation of a poorly conditioned system with λ weighting set to 1.0 55
3.8 Simulation of a poorly conditioned system with λ weighting set to 1.0 56
3.9 Simulation of a poorly conditioned system with λ weighting set to 20.0 and $\gamma_{1,2,3,4,5}$ = 1.0,0.5,0.5,0.5,0.5. 58
3.10 Simulation of a poorly conditioned system with λ weighting set to 20.0 and $\gamma_{1,2,3,4,5}$ = 1.0,0.5,0.5,0.5,0.5. 59
3.11 The effect of precision of variables on open loop simulations 60
3.12 The effect of precision of variables on open loop simulations 61

4.1 Schematic of Shell heavy oil fractionator 66
4.2 Block diagram of heavy oil fractionator 66

4.3 Step response matrix of the Shell Control Problem 70
4.4 Choosing initial tuning parameters for GPC 73
4.5 Choosing initial tuning parameters for GPC 74
4.6 Closed loop simulation without autotuning and output constraints 77
4.7 Closed loop simulation without autotuning and output constraints 78
4.8 Closed loop simulation with tuned output weighting 80
4.9 Closed loop simulation with tuned output weighting 81
4.10 Closed loop simulation with tuned T-filter 83
4.11 Closed loop simulation with tuned T-filter 84
4.12 Closed loop simulation with output constraints 85
4.13 Closed loop simulation with output constraints 86
4.14 Closed loop simulation with output constraints and tuned output weighting . 87
4.15 Closed loop simulation with output constraints and tuned output weighting . 88
4.16 Closed loop simulation with MPM, tuned output weighting and T-filter 91
4.17 Closed loop simulation with MPM, tuned output weighting and T-filter 92
4.18 Closed loop simulation with MPM and tuned T-filter 93
4.19 Closed loop simulation with MPM and tuned T-filter 94
4.20 Simulation of a 5-input 4-output system with all 5 inputs active 97
4.21 Simulation of a 5-input 4-output system with all 5 inputs active 98
4.22 Simulation of a 5-input 4-output system with 4 inputs active 99
4.23 Simulation of a 5-input 4-output system with 4 inputs active 100
4.24 Simulation of a 5-input 4-output system with saturation detection 101
4.25 Simulation of a 5-input 4-output system with saturation detection 102

5.1 Schematic of the Continuous Stirred Tank Heater 108
5.2 Hardware configuration of the equipment 111
5.3 Effect of output weighting on closed loop response 115
5.4 Data transmission and program configuration 118
5.5 Experimental evaluation of GPC with autotuner disabled 121
5.6 Experimental evaluation of regulatory supervision 123
5.7 Experimental evaluation of servo supervision 124

5.8 Results with regulatory and servo supervision enabled 126
5.9 Results with regulatory and servo supervision enabled 127

Chapter 1

Introduction

The evolution of control strategies has been progressive, from Proportional Integral and Derivative (PID) controllers for single-input, single-output (SISO) systems, to sophisticated multi-input, multi-output (MIMO) controllers such as the Generalized Predictive Control or GPC (Clarke *et al.*, 1987). PID control loops are used extensively in industry, due to their simple mathematical formulation and easy configureability.

Over the years, other control strategies such as the Dynamic Matrix Control or DMC algorithm of Cutler and Ramaker (1979), Multivariable, Optimal, Constrained Control Algorithm (MOCCA) of Sripada and Fisher (1985) and GPC have attracted research and industrial attention due to their ability to enhance closed loop performance by considering multivariable processes as complete entities for control systems design, in contrast with the 'independent or multi-loop design' method of conventional PID controllers.

Commissioning and tuning a control system is an important milestone before a plant can be considered fully operational. There are a number of methods for tuning PID loops for single-input, single-output or SISO plants and multi-input, multi-output or MIMO plants. The task of tuning PID controllers is often non-trivial especially if excellent tracking and load rejection characteristics are desired. Also, due to interactions present in MIMO systems, it becomes even more important to choose proper PID controller parameters because an incorrect choice can result in an unstable closed loop system.

It was only recently that the idea of using expert systems to configure and tune PID loops became of interest to many researchers. Expert systems are domain specific, problem solving programs which can incorporate human knowledge and heuristics in the knowledge base. A knowledge base is a collection of heuristics and/or numerical algorithms used to solve a specific problem. A few recent research papers in developing expert systems for SISO PID controllers are by James *et al.* (1987), Lewin and Morari (1988) and Thomson *et al.* (1989). A few contributions to research in MIMO expert systems are due to: Boyle *et al.* (1989) with their Multivariable Analysis and Interactive Design of control systems or MAID expert system, Pang (1992) for an updated version of MAID and Tebutt (1992) for designing an interactive expert system which assists the user in determining achievable design specifications for a system with up to 5 inputs and 5 outputs.

Tuning the 'newer' control algorithms such as DMC, MOCCA and GPC requires knowledge and an intimate understanding of these algorithms, which at least as of now is still not easily available in most plant environments. In order to assist a plant operator/engineer off-line, an expert system based tuning method can be devised for DMC, MOCCA or GPC. Besides being used as a decision support system, such an expert system can also be used for operator training. One such expert system called ALPS (Adaptive Long range predictive control Performance Supervisor) developed by Walther (1992) is used to tune GPC on a 2-input 2-output pilot plant. The ALPS expert system tunes selected GPC tuning parameters and is found to work well in the presence of noise.

This thesis is part of an ongoing research effort to develop a comprehensive tuning strategy for GPC.

1.1 Issues in multivariable control

In order to understand how to control multivariable systems, it is necessary to have some insight into the dynamics and structure of the process. Sometimes even a little knowledge of the process, e.g. that the system contains gains which are orders of magnitude apart, can help in making a good decision about initial tuning parameters in the control strategy of choice.

A transfer function representation of a 3-input, 3-output process is given by:

$$\begin{bmatrix} y1 \\ y2 \\ y3 \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} & G_{13} \\ G_{21} & G_{22} & G_{23} \\ G_{31} & G_{32} & G_{33} \end{bmatrix} \begin{bmatrix} u1 \\ u2 \\ u3 \end{bmatrix} \tag{1.1}$$

The complexity of the above multivariable systems can be illustrated by Figure 1.1. This figure shows the interactive nature of a 3-input, 3-output PID control strategy. Since the output of each controller affects all three process outputs, it becomes very difficult to "pair" an input and an output. Even if "pairing" of inputs and outputs is achieved, tuning the control loops can be extremely difficult due to the process interactions.

It is thus necessary to develop a control strategy by considering the process as an entity, rather than on a loop by loop basis. A multivariable, model-based controller implicitly considers all the interations when calculating the control trajectory. There are a number of model-based control algorithms such as Pole-placement and Long Range Predictive Control (LRPC). The Generalized predictive control algorithm or GPC, belongs to the larger family of LRPC algorithms and will be the control strategy of choice in this work.

1.2 Expert GPC

It is not very obvious how to tune GPC for 'good' performance. Certain theoretical results for tuning single-input, single-output (SISO) GPC systems have been developed by McIntosh *et al.* (1991). The paper by Mohtadi *et al.* (1991) talks of tuning guidelines for MIMO GPC, but an exhaustive study in this area has not yet been done. Simulations carried out by Mutha (1990) suggest a method by which initial tuning parameters can be chosen and how a 3-input, 3-output process can be tuned by trial and error. The ALPS

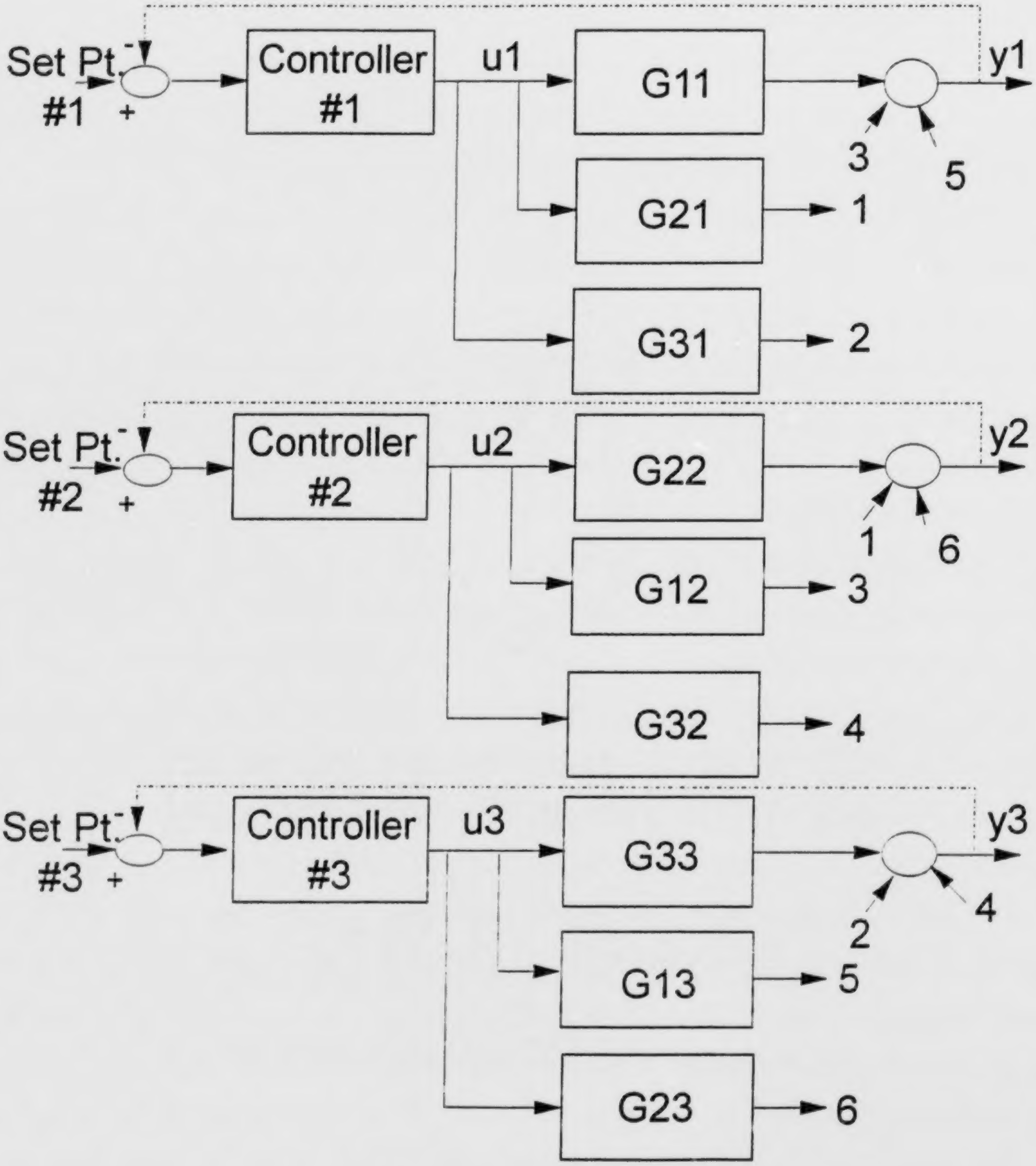


Figure 1.1 Block diagram of a 3-input 3-output MIMO system with three SISO feedback controllers

expert system tuning strategy of Walther (1992) was evaluated on a 2-input 2-output process only, which is not representative of the larger problems encountered in industry. Also, ALPS was evaluated in simulation mode only. The purpose of this work is to apply and extend ALPS to larger multivariable systems, *e.g.* with 5 inputs and 5 outputs, and to evaluate it experimentally on a 2-input 2-output process.

1.3 Objectives of study

The objective of this work is to gain some insight into the behaviour of large processes with complex interactions and develop an appropriate tuning strategy for GPC. These investigations were carried out by doing the following:

- Extending the knowledge base developed by Walther (1992) to a maximum of 5 inputs and 5 outputs. This knowledge base is used in tuning the GPC parameters.
- Analyzing the dynamics of large multivariable systems and identifying characteristics which help in tuning GPC.
- Developing software for communicating with an experimental process. The experimental process is a 2-input 2-output system located remote from the computer.
- Evaluating ALPS on the experimental process.
- Analyzing the closed loop response of the constrained GPC algorithm when some inputs in an overspecified system, for example, a 5-input, 4-output problem, get saturated.

1.4 Structure of thesis

This work is organized in a way so as to first provide the reader with a brief background in the area of research and then highlight the contributions made by the author.

An introduction to the Generalized Predictive Control algorithm, GPC, and the idea of process supervisory systems is discussed in chapter 2. The construction of the knowledge base is not the main subject of discussion, however some modifications and additions made to the ALPS knowledge base by the author, to account for problems of larger dimensions, are discussed. In chapter 3 a brief introduction to the structure of multivariable systems and their level of complexity is discussed. Certain issues which affect the choice of some design parameters are discussed here as well. The performance of the GPC algorithm is evaluated by simulations using a 5-input, 5-output benchmark problem in chapter 4. Different sets of simulations are carried out to develop and justify the choice of *initial* tuning parameters of GPC. Autotuning simulations are also a part of this chapter. Since simulations cannot duplicate all the characteristics of a real process, experimental evaluation of the GPC autotuner is considered in chapter 5. The experimental equipment is a 2-input, 2-output process which is remote from the computer. A number of experiments are performed to test the validity and robustness of the autotuner. Chapter 6 draws conclusions from previous chapters and possibilities for future work are discussed.

Chapter 2

Expert Generalised Predictive Control

The focus in this chapter is on the characteristics of the Generalised Predictive Control (GPC) algorithm and the method used to devise an autotuning strategy for this algorithm. The GPC algorithm was proposed by Clarke *et al.* in 1987. This algorithm belongs to the larger family of Long Range Predictive Control (LRPC) algorithms of which the Dynamic Matrix Control (DMC) algorithm (Cutler and Ramaker, 1979) is also a member. Sections 2.1 and 2.2 provide background information about the characteristics of the LRPC and the GPC algorithms respectively. A review of work done on expert systems is presented in section 2.3. Knowledge based systems and their use and justification in autotuning are discussed in section 2.4. The structure of the knowledge base and tuning criteria are contained in section 2.5 followed by conclusions in section 2.6.

2.1 Long Range Predictive Control

The objective of Long Range Predictive Control (LRPC), is to minimize a suitable function of the errors between the set point or desired value of an output and the predicted output, over a finite (user-specified) time horizon. This idea can be illustrated graphically in figure 2.1. In this figure, the known past inputs and outputs are shown, together with projected outputs and inputs. From time t to t+N, which is a window of N sample instances, the projected error between the output, and its set point is minimized.

The minimization is done by using the past inputs and outputs and an objective function. The objective function can be a quadratic cost function, in which case the sum of the squares of the errors is minimized, or a linear objective function, in which case the sum of the errors is minimized. Other user defined cost functions can be used as well. The quadratic objective function is most commonly used because minimizing this objective function results in minimizing the energy cost to a process, which is of practical importance. Another reason for using the quadratic cost function is that an analytical solution of the resulting unconstrained optimization exists. A block diagram of the LRPC strategy is presented in figure 2.2. The different aspects of this block diagram are:

The control law: The control law is the equation which is derived after the objective function mentioned earlier is minimized with respect to u, the process input.

The process: This is the actual process unit which exists and is to be controlled. In the case of simulations, the process is replaced by a model. In the case of DMC, a step response model is used but a mathematical model (ARIMAX) is used for GPC. This is the only major difference between DMC and GPC, since both belong to the LRPC family.

The predictor: The prediction of the output is based on the previous values of the input and output plus a suitable model.

Details of LRPC are available in the M.Sc. thesis by Walther (1992). Since the concepts of GPC are identical to those of LRPC, all relevant equations are presented in the following section on GPC. The discussion on DMC is omitted as this work is concerned mainly with tuning aspects of GPC. Details of the DMC algorithm can be found in the paper by Cutler and Ramaker (1979). Some features of the LRPC strategy are as follows:

- The LRPC strategy can be applied to multivariable processes. This means that large process units can be considered as an entity and a control system designed for the whole unit rather than a number of individual process variables. Also, large multi-input multi-output processes can be handled by this control strategy. Increasing the size of the application does increase the demand on computer resources, but with easy access to powerful computer systems, this is not a real

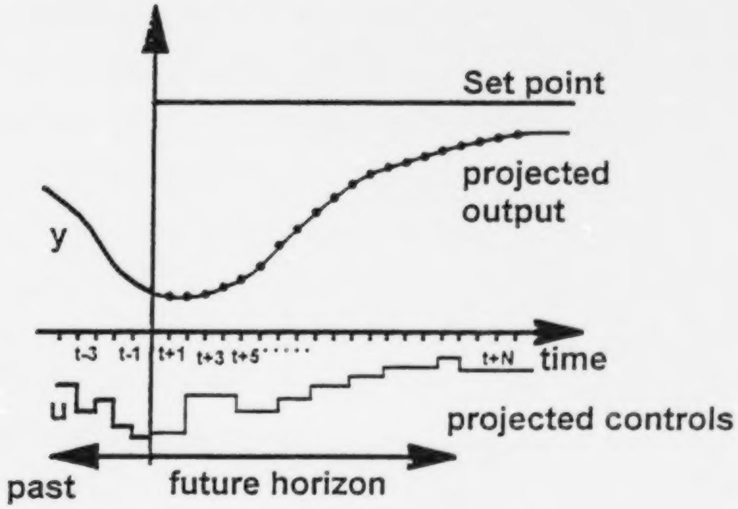


Figure 2.1 The concept of long range prediction

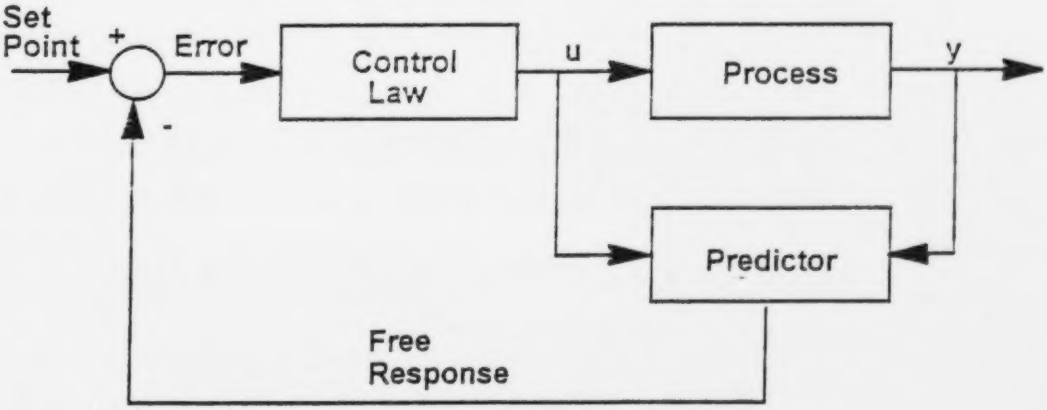


Figure 2.2 Block diagram representation of LRPC

concern.

- The quadratic cost function of LRPC leads to an optimal control trajectory such that the square of the error between the actual and predicted outputs is minimized. This means that a globally optimal operating condition can be found.
- It is very easy to include constraints in the problem formulation of LRPC. Since hardware constraints for safety reasons are very common in industry, this is a very useful and practical feature of LRPC. The unconstrained solution of the LRPC algorithm is analytical, but the constrained solution of the LRPC algorithm requires the use of a solution searching algorithm.

With all these features, LRPC looks attractive for use as a control algorithm. Specifically, GPC is considered for this work over DMC due to fewer parameters in the model representation of GPC.

2.2 The Multivariable GPC algorithm

GPC is a true multivariable controller. Unlike many other frequency-domain based multivariable techniques (such as Direct Nyquist Array or DNA, Inverse Nyquist Array or INA, Characteristic Loci or CL etc.), it is a quadratic cost-function based technique where interaction compensation is achieved through specification of the appropriate output weighting parameters (Mohtadi *et al.*, 1991). In this section some equations and results are presented to illustrate the theory behind GPC and the parameters involved in tuning the closed loop performance to a desired specification. The equations presented here are valid for an n-output, m-input multivariable process. Details of the GPC algorithm can be found in Mohtadi *et al.* (1991) and Mutha (1990).

2.2.1 The process model

There are a number of ways of representing the process model. The model considered for this work is the Auto Regressive Integrated Moving Average with eXogenous noise or ARIMAX model. For an n-output, m-input process, this model can be represented as follows:

$$Ay(t) = Bu(t-1) + \frac{C}{\Delta}e(t) \tag{2.1}$$

where,

A and C are diagonal polynomial matrices of dimension n by n.

B is a polynomial matrix of dimension n by m.

y and u are the output and input vectors of dimension n by 1 and m by 1 respectively.

e is a noise vector of dimension n by 1 having zero mean.

Δ is the differencing operator, $\Delta = 1\text{-}q^{-1}$.

The diagonal elements of the A matrix can be written as:

$$A_{ii}(q^{-1}) = 1+a_1q^{-1}+\ldots+a_{\delta A_{ii}}q^{-\delta A_{ii}} \tag{2.2}$$

where: δA_{ii} represents the order of the i^{th} element. i = 1,2, ... n.

Similarly, the elements of the B matrix can be written as:

$$B_{ij}(q^{-1}) = 1+b_1q^{-1}+\ldots+a_{\delta B_{ij}}q^{-\delta B_{ij}} \tag{2.3}$$

where: δB_{ij} represents the order of the ij^{th} element. i = 1,2, ... n and j = 1,2 ... m.

If any of the input-output channels of the process have a non-zero time delay, then the leading coefficients of the corresponding polynomial in the B matrix are zero.

2.2.2 The objective function

As discussed earlier, all the LRPC algorithms have the ability to calculate an optimum control trajectory based on an objective function. The objective function used in this work is the quadratic objective function. The multivariable GPC objective function to be minimized is given by:

$$J_{GPC} = \sum_{j=N1}^{N2} [\hat{y}(t+j)-w(t+j)]^T \Gamma \, [\hat{y}(t+j)-w(t+j)] + \sum_{j=1}^{NU} [\Delta u(t+j-1)^T \Lambda \, \Delta u(t+j-1)] \tag{2.4}$$

where,

$\hat{y}(t+j)$ is a vector of predicted outputs of dimension n by 1.

$w(t+j)$ is a vector of future set points of dimension n by 1.

$\Delta u(t+j)$ is the input vector of dimension m by 1.

N1 is the minimum (or start of) output prediction horizon.

N2 is the maximum (or end of) output prediction horizon.

NU is the control horizon.

Γ is a diagonal output weighting matrix of dimension n by n. A different output weighting for each output can be specified as shown in the following equation.

$$\Gamma = \begin{bmatrix} \gamma_1 & & \\ & \ddots & \\ & & \gamma_n \end{bmatrix} \tag{2.5}$$

One method of choosing output weighting is to implement maximum output weighting on the output which deviates most from its set point, and scale the weighting on the other outputs accordingly. Mutha (1990) in his work concluded that output weighting should be selected based on the amount of model plant mismatch between the actual plant and the model *e.g.* if the gain of a transfer function which affects output y1 contains an estimated mismatch of 30% as compared to a mismatch of 20% in the gain of a transfer function which affects y2, then the output y1 should have higher output weighting than output y2.

Λ is a diagonal control weighting matrix of dimension m by m. A different control weighting for each input can be specified as shown in the following equation.

$$\Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_m \end{bmatrix} \tag{2.6}$$

A weighting on the inputs is used to control the amount of movement produced in a certain input. The higher the control weighting, the smaller will the movement be in that input. It is also known as the move suppression parameter. Control weighting can also be used to conserve costs in a real plant by having a large control weighting on expensive inputs *i.e.* it can be used as a control penalty parameter. Other effects of the control weighting are discussed in section 2.2.4

2.2.3 Long range prediction of process outputs

In order to minimize the predicted error between the process outputs and their set points, as given by equation 2.4, the predicted process output $\hat{y}$ needs to be calculated. This output is given by equation 2.7. It should be noted that the prediction equations for DMC and GPC (for the integrated white noise case) are identical, since both DMC and GPC are long range predictive control algorithms.

$$\hat{y} = G_d \, \Delta u + f \tag{2.7}$$

where,

$\hat{y}$ is the predicted output vector of dimension (N2-N1+1)*n by 1. This output vector includes the impact of future control actions.

Δu is the input vector over the entire control horizon. The dimension of Δu is (m*NU) by 1, *i.e.* NU input vectors each of dimension m are calculated.

f is the free response of the process and is of dimension (N2-N1+1)*n by 1 and consists of the predicted process outputs assuming no future changes in the control action. There are two methods of obtaining the free response of the process (Mutha, 1990):

- By using the Diophantine identity which is given by

$$I = E_j A\Delta + q^{-j} F_j \qquad (2.8)$$

where E_j and F_j are matrices of Diophantine coefficients. In practice, E_j and F_j are obtained by multiplying both sides of equation 2.8 by $(A\Delta)^{-1}$ (or dividing by $A\Delta$). By doing this, E_j is the quotient matrix and $F_j(A\Delta)^{-1}$ is the remainder matrix. Furthermore, E_j consists of the step-response coefficients of the MIMO process. An on-line implementation of the Diophantine identity requires redimensioning of E_j and F_j whenever the values of N1 and/ or N2 are changed.

- Setting Δu equal to zero in equation 2.7 will produce $\hat{y} = f$, which means that the predicted output and the free response are the same thing when future control actions are set to zero. A recursive method can be used in which future values of $\Delta u(t+j)$ are set to 0 and the free response is calculated by using the model in equation 2.1.

The recursive method is preferred over the Diophantine identity method because it is easier from a software implementation point of view and the recursive method does not require any permanent storage of data, unlike the Diophantine identity method which requires permanent storage.

G_d is the dynamic matrix of the process and has a dimension of (N2-N1+1)*n by (m*NU). The structure of this matrix is as follows:

$$G_d = \begin{bmatrix} G_{N1-1} & \cdots & G_0 & 0 & 0 & \cdots & 0 \\ G_{N1} & \cdots & G_1 & G_0 & 0 & \cdots & 0 \\ \vdots & \ddots & & & & & \vdots \\ G_{N2-1} & \cdots & & & & & G_{N2-NU} \end{bmatrix} \qquad (2.9)$$

Each G_i in equation 2.9 above is a submatrix of dimension n by m, the elements of which are the step response coefficients of the i^{th} sampling interval of each individual input-output relationship.

2.2.4 The multivariable GPC control law

The objective function in equation 2.4 is minimized with respect to u by taking partial derivatives and setting them to 0. This exercise results in the following control law:

$$\Delta u(t) = [I_m \; 0 \; \cdots \; 0] \, [G_d^T \underline{\Gamma} \, G_d + \underline{\Lambda}]^{-1} \, G_d^T \underline{\Gamma} (w-f) \qquad (2.10)$$

where,

$[I_m \; 0 \ldots 0]$ is a pre-multiplication vector. This vector is required because only the first control increment is implemented, the rest are used for further calculations only.

G_d is the dynamic matrix given by equation 2.9.

$\underline{\Gamma}$ is a matrix of (N2-N1+1) block diagonal Γ matrices (see equation 2.5 for Γ matrix).

$\underline{\Lambda}$ is a matrix of NU block diagonal Λ matrices (see equation 2.6 for Λ matrix).

w is the future set point trajectory given by

$w = [w(t+N1) \ldots w(t+N2)]^T$

The dimension of the above vector is (N2-N1+1)n by 1.

f is the free response of dimension (N2-N1+1)n by 1 and is given by

$f = [f(t+N1) \ldots f(t+N2)]^T$

The matrix represented by $[G_d^T \underline{\Gamma} \, G_d + \underline{\Lambda}]$ is known as the *Hessian*. In this control law, it is important to note that the inverse of the *Hessian* matrix has to be calculated. If the *Hessian* is poorly conditioned, that is, two or more columns of the matrix are linearly dependent or almost linearly dependent, then the inverse will not exist or be difficult to calculate and this will result in incorrect calculation of the control action vector $\Delta u(t)$. Since the only variables which effect the *Hessian* are the input weighting vector Λ and the output weighting factor Γ, this means that manipulating the values of Λ or Γ can make an otherwise poorly conditioned system well conditioned.

2.2.5 Extensions to the multivariable GPC algorithm

Although the fundamental algorithm is as presented in the preceding sections, certain modifications to this algorithm are necessary to make it robust to noise and disturbances. Also, the idea of incorporating hard constraints into the problem formulation of GPC is presented in this section.

2.2.5.1 Use of the T-filter to enhance robustness

The noise term in equation 2.1 is given by $Ce(t)/\Delta$, where e(t) is zero-mean white noise. Multiplying e(t) by C/Δ results in integrated (non-stationary) coloured noise. Most process noise is of this type and therefore this is a fairly realistic representation. However, obtaining estimates of the diagonal polynomial matrix C for a real process is difficult, thus a fixed polynomial matrix, the T matrix, is used instead of the C matrix. Different $T(q^{-1})$ polynomials can be specified for different outputs. In practice a first or second order $T(q^{-1})$ polynomial (also referred to as the T-filter) is sufficient for good noise and disturbance rejection characteristics.

Using the T-filter results in the following modified Diophantine equation:

$$T = E_j\, A\Delta + q^{-j}F_j \qquad (2.11)$$

When equation 2.11 is used with the process model (equation 2.1), the following modified prediction equation is produced:

$$\hat{y}_f = G_d\Delta u_f + f_f \qquad (2.12)$$

All quantities with the subscript $(\ )_f$ in equation 2.12 represent filtered vectors. The calculation of the filtered free response f_f is the same as the calculation of the free response, except that filtered values of $\hat{y}$ and Δu are required rather than the unfiltered values as described in section 2.2.3. Since the control law in equation 2.10 does not require the filtered free response, f_f but the unfiltered free response, f, this means that *inverse* filtering has to be carried out as follows:

$$f = f_f\, T \qquad (2.13)$$

For the single input single output case, the reader is referred to Clarke (1991) for further information on *inverse* filtering.

2.2.5.2 Feedforward compensation

Since real processes are prone to disturbances which degrade closed loop performance, disturbance rejection is a very important issue in process control. Sometimes a model of the disturbance is available. This information can be used to enhance disturbance rejection characteristics of GPC by including the model of the disturbance in the process model described in equation 2.1. This modified model is used to calculate the free response of the process, but the step response matrix is calculated by using the uncompensated, original model. This method of compensating for disturbances is analogous to the PID feedforward technique in which the knowledge of the model of the disturbance is used to compensate for the disturbance before it has an effect on the process, thus the name 'feedforward compensation'. Further details of this method are available in Mutha (1990).

2.2.5.3 Input and output constraints

The absolute control action calculated from the control law in equation 2.10 can have negative or positive values. In reality however, control valves can only operate within 0% and 100% open, requiring the use of constraints on the absolute amplitude of the control action. Rate constraints on inputs and absolute or amplitude constraints on outputs can also be imposed.

1. Rate constraints:

Rate constraints can directly be incorporated in the problem formulation of GPC. For a general multivariable process, rate constraints can be represented as follows:

$$\begin{bmatrix} \Delta u_{\min,1} \\ \Delta u_{\min,2} \\ \vdots \\ \Delta u_{\min,m} \end{bmatrix} \le \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+NU-1) \end{bmatrix} \le \begin{bmatrix} \Delta u_{\max,1} \\ \Delta u_{\max,2} \\ \vdots \\ \Delta u_{\max,m} \end{bmatrix} \qquad (2.14)$$

Where m is the number of inputs. Rate constraints are user specified and depend on the sampling time of the system.

2. Input amplitude constraints:

Amplitude constraints can be represented in a similar method as in equation 2.14,

$$\begin{bmatrix} u_{\min,1} \\ u_{\min,2} \\ \vdots \\ u_{\min,m} \end{bmatrix} \le \begin{bmatrix} u(t) \\ u(t+1) \\ \vdots \\ u(t+NU-1) \end{bmatrix} \le \begin{bmatrix} u_{\max,1} \\ u_{\max,2} \\ \vdots \\ u_{\max,m} \end{bmatrix} \qquad (2.15)$$

however they have to be translated into rate constraints.

This is done as follows:

we know that $\Delta u(t) = u(t) - u(t-1)$, thus equation 2.15 can be written as:

$$\begin{bmatrix} u_{\min,1}-u(t-1)_1 \\ u_{\min,2}-u(t-1)_2 \\ \vdots \\ u_{\min,m}-u(t-1)_m \end{bmatrix} \le \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+NU-1) \end{bmatrix} \le \begin{bmatrix} u_{\max,1}-u(t-1)_1 \\ u_{\max,2}-u(t-1)_2 \\ \vdots \\ u_{\max,m}-u(t-1)_m \end{bmatrix} \tag{2.16}$$

The dimension of the vectors are (m*NU) by 1 and the dimension of the lower diagonal matrix is (m*NU) by (m*NU).

3. Output amplitude constraints:

Output amplitudes can be represented by the following equation:

$$y_{\min} \le \hat{y} \le y_{\max} \tag{2.17}$$

substituting equation 2.7 in equation 2.17 gives:

$$\begin{aligned} y_{\min} &\le G_d \Delta u + f \le y_{\max} \\ \rightarrow\ y_{\min}-f &\le G_d \Delta u \le y_{\max}-f \end{aligned} \tag{2.18}$$

The vectors $y_{\min}$, $y_{\max}$, f and Δu are dimensioned depending on the values of N1, N2 and NU.

The solution of the constrained multivariable GPC problem is obtained by minimizing the objective function in equation 2.4 subject to constraints 2.14, 2.16 and 2.18. There is an analytical solution available (Mutha, 1990) to a m-input, n-output MIMO system with constraints for NU=1. For the general case of NU $\neq$ 1, there is no analytical solution, and the resulting optimization problem can be solved via a commercial software. A commercial quadratic optimization software package called QPSOL (Gill *et al.*, 1984) is used to solve the constrained optimization problem on-line at every sampling interval. The commercial software consists of about 6,000 lines of ANSI-66 FORTRAN, and can be implemented using single precision or double precision variables.

2.3 Related work

Expert systems consist of domain specific knowledge from a human expert(s), encoded in a computer program. This program is then used to solve domain specific problems. Benefits generated by the use of expert systems technology have resulted in a growing research interest in this area.

The PICON expert system development environment (Moore *et al.*, 1984; Moore and Kramer, 1986) was the predecessor of G2 (Moore *et al.*, 1988). The G2 expert system development environment was designed by keeping practical issues such as large number of process measurements, combination of numerical and symbolic processing, in mind. A few applications of Gensym's G2 are due to Nevins (1989), Aynsley *et al.* (1989) and Braunstein *et al.* (1989). In all these applications, G2's real-time clock, object-oriented generic representation of variables, interactive rule development and icon based graphical interface are cited as valuable features. This idea of encapsulating knowledge in a generic, higher level of representation is discusssed by Årzén (1989), Alty and Johannsen (1989) and Pang (1992). They conclude that the generic and high level representation features are very useful, and almost essential, for future developments and modifications to the knowledge base of an expert system.

The choice of G2 is thus appropriate for this work which involves simulation of a large 5-input 5-output process in real-time. The features of G2 make it easy to develop and modify rules and allows for mathematical formulas and heuristics to be represented as part of the same knowledge base.

James *et al.* (1987), Lewin and Morari (1988) and Thomson *et al.* (1989) discuss implementing lead-lag compensators, designing a robust PI or IMC controller using the ROBEX expert system and tuning PID controllers, respectively. However, all these applications are limited to single-input single-output processes only. The MAID expert system (Boyle *et al.*, 1989) can design PI controllers for linear MIMO systems by interacting with the designer, but this expert system has a serious limitation that it cannot handle delays in the process. The expert system discussed in this work is applicable to MIMO systems with delays, which is an improvement over the expert systems mentioned above.

From the above discussion, it is quite clear that the G2 environment is very user-friendly and efficient for developing a knowledge base for an expert system. Combining this environment with a multivariable control algorithm such as GPC (see section 2.2) can thus be powerful in solving large and complex process control problems.

2.4 Expert GPC

There are a number of autotuners available for PID type controllers. Two particular examples are the autotuner by Toshiba and Foxboro's Expert Adaptive Controller (EXACT). While the Toshiba controller automatically sets values of the Proportional, Integral and Derivative variables once and keeps them constant, the EXACT controller updates the PID variables following a set point change in the process. In both cases, the PID constants are conservative and may need to be tuned further to obtain a faster response.

The use of expert systems for tuning a control algorithm comes naturally, since most industrial plants are operated by people who have experience with the plant and operate it based on their knowledge. Expert systems can capture this knowledge into a computer program and use it to assist the operator in day to day operations. Expert systems are also very valuable in passing on this expertise to other people in a systematic and organized way, so that a novice operator can be trained easily. Once an expert system is implemented, the knowledge base, which contains all the knowledge of the domain expert, can be enhanced and updated by experience of other people.

The focus in this section is on the expert supervisor used for tuning the GPC algorithm. The name of this expert GPC program is Adaptive Long range predictive control Process Supervisor or ALPS. Only the main concepts and highlights of ALPS are presented in this section. Details of ALPS are available in the M.Sc. thesis by Walther (1992). It should be noted that the terms "ALPS" , "expert GPC" and "expert system supervisor" are used synonymously and interchangeably in this thesis.

2.4.1 Justification for developing Expert GPC

Expert systems allow the tuning experience of even one expert to be spread to a number of other plant personnel, without the expert being physically present. As mentioned in the section above, using expert systems for tuning a control algorithm comes naturally. This is even more so for the tuning of a long range predictive control algorithm such as GPC due to the lack extensive experience in tuning this algorithm and the lack of an exhaustive (theoretical) multivariable closed loop analysis of the effect of various tuning parameters on the closed loop response.

The tuning strategy used in this work is based on information from the process and *rules* derived from tuning strategies for GPC for single-input single-output (SISO) processes. A detailed description of the different tuning strategies of GPC is given by Walther (1992). The focus in this work is on a generic tuning strategy for large multivariable processes, and an experimental evaluation of ALPS, the closed loop performance supervisor. Possible functions of a process supervisor are presented graphically in figure 2.3.

2.4.2 The architecture of expert GPC

Expert systems are a subset of knowledge based systems, which in turn are a subset of the larger field of artificial intelligence. The development of an expert system consists of encoding the domain specific knowledge of an individual into a computer program. This program can then be used to solve domain specific problems, train other individuals and most importantly, this program can be modified to include the knowledge and expertise of other domain experts.

Making an expert system on-line requires extensive error checking and knowledge base validation so that incorrect tuning parameters are not implemented due to logical or numerical errors. The hierarchy of the expert system is presented in figure 2.4. This figure shows a top down approach, with the expert system supervisor, ALPS, at the very top. The control algorithms and ALPS are housed in the same computer but this is not representative of the norm and the two may be resident on different computers. Since the control algorithms and the expert system supervisor operate independently, an extra degree of fault tolerance or safety is achieved. This also results in the best utilisation

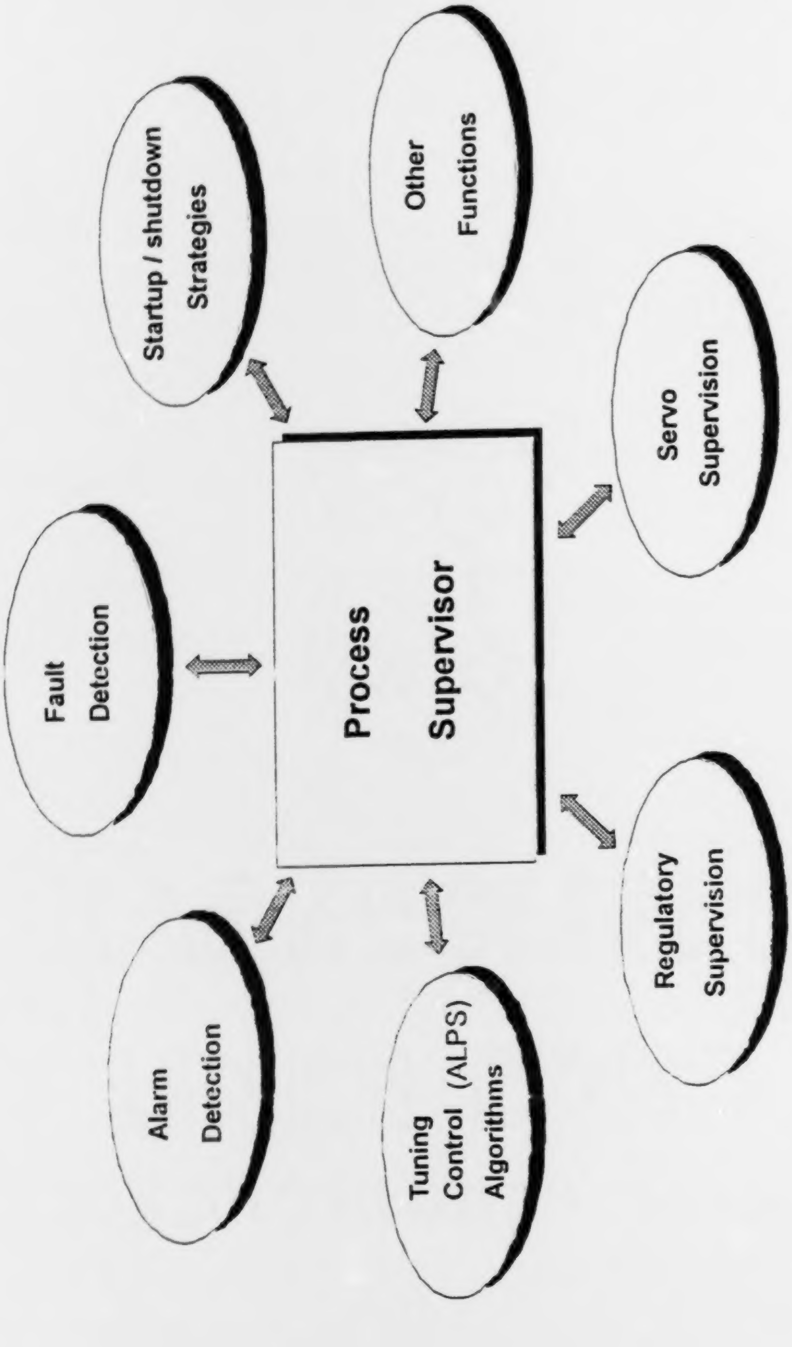


Figure 2.3 Possible functions of the process supervisor

of expert systems technology which is symbolic processing, combined with the fast numerical processing capabilities of the control algorithms. Further down the hierarchy is the computer / process interface and finally the instrumentation and the process. For simulation purposes, the process is replaced by a simulation, but for an experimental evaluation of ALPS, the remaining part of the hierarchy is relevant as well. Details of the experimental evaluation of ALPS are presented in chapter 5.

2.4.3 Choice of hardware and software

In order to provide some perspective on the choice of hardware, it is relevant to point out the size of the computer programs being executed in this application of ALPS. Table 2.1 gives approximate sizes of the programs and the programming language used. The multivariable GPC program consists of a number of functions written in the 'C' programming language. The G2 Standard Interface (GSI) is a program which provides a one to one correspondence between the variables in the ALPS knowledge base and external programs such as the multivariable GPC programs. The QPSOL software is the commercial optimisation software which is used to solve the constrained GPC problem. The ALPS knowledge base is constructed by using the high level language of G2, but G2 itself is LISP based.

For executing all this software in real-time in an on-line environment (for the experimental application) required a fast computer with capabilities of compiling mixed language programs, thus the HP-9000 computer was chosen as the hardware platform. The HP-9000 workstation uses the HP-UX real-time multitasking disk operating system. Three 300 series computers, the models 320, 340 and 370 are located on thesame cluster, of which the model 370 is the disk server for the other two computers. The model 370 workstation uses the motorola 68030 microprocessor and operates at a clock speed of 33 MHz.

Gensym's G2 expert system shell is used for the knowledge base development because of the object oriented programming capabilities of this shell, which allow for the development of generic rules, applicable to any plant, thus making the program portable. The G2 shell also has a real time clock, thus simulations can be done

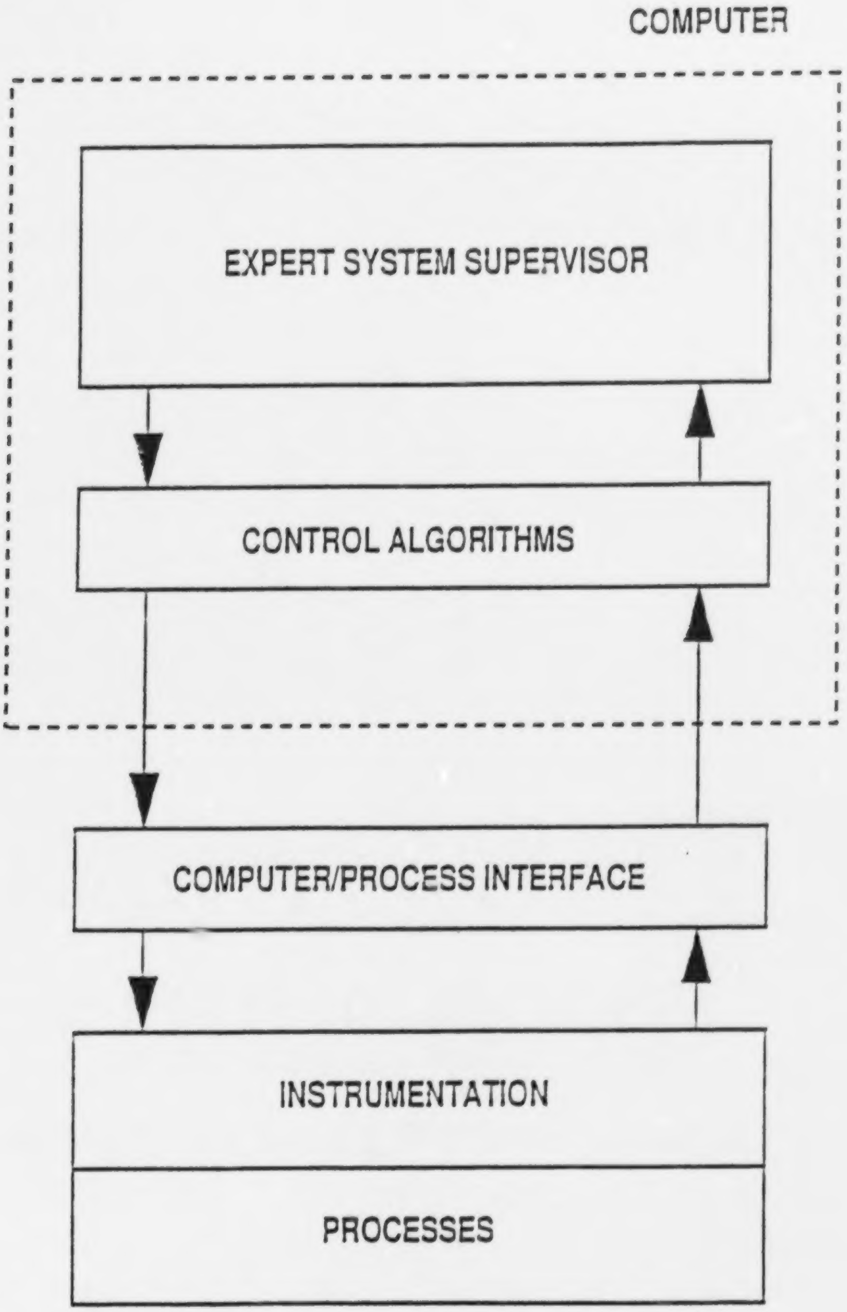


Figure 2.4 Hierarchy of the expert systems supervisor

in real-time. The real-time clock is very useful for an experimental application because

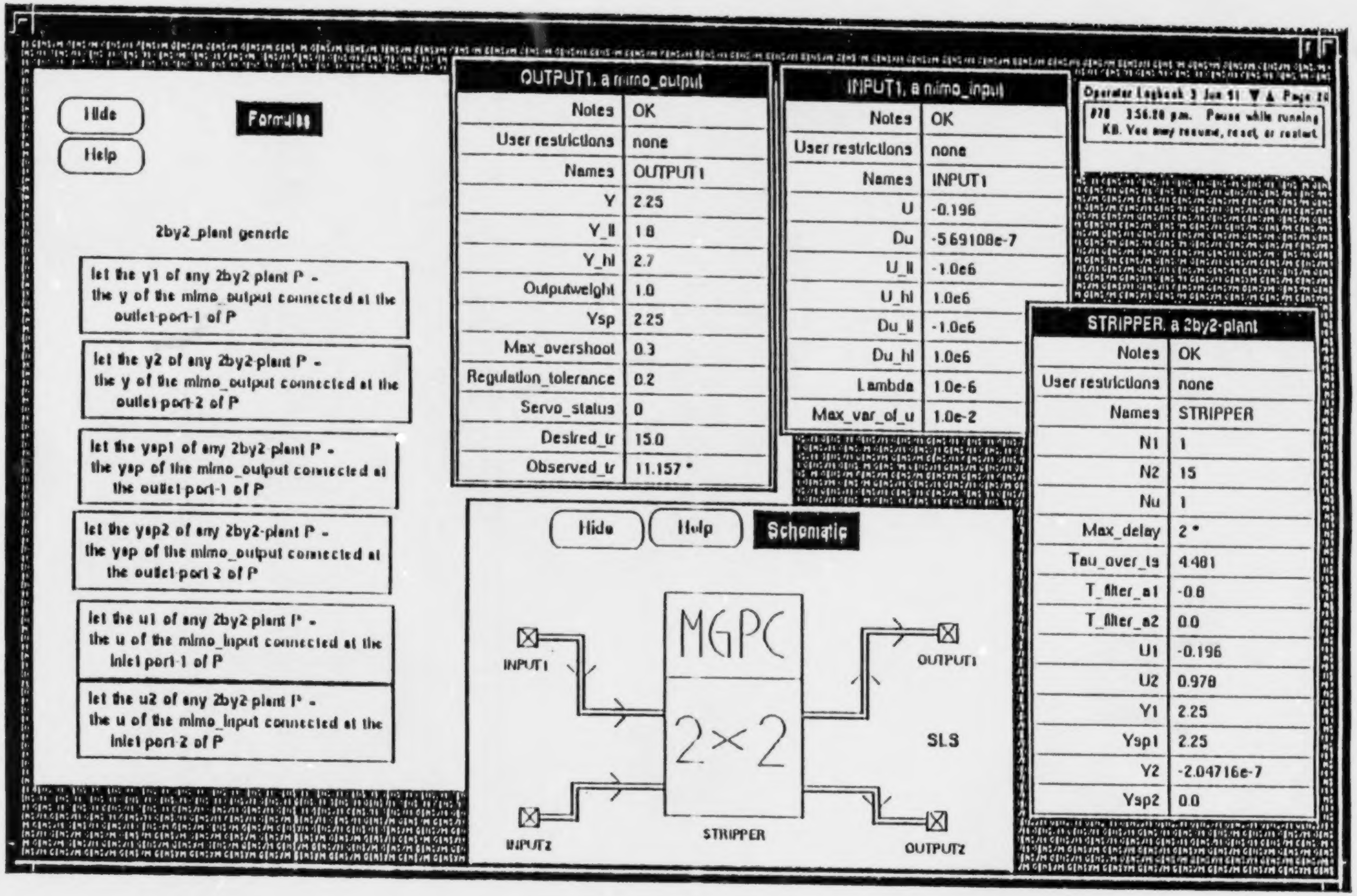
Program	Language	# of lines of code
Multivariable GPC	C	3500
G2 Standard Interface (GSI) Link	C	1500
QPSOL	FORTRAN	6000
ALPS knowledge base	LISP-based high level language on G2	3000

Table 2.1 A summary of the source code executed

of the ease in which the sampling time can be specified. Also, the nice windowed environment of G2 makes it very user friendly and easy to develop a knowledge base. An example of the windowed environment is presented in figure 2.5. The rules written contain virtually no syntax errors due to the english language editor which prompts the user with a choice of keywords when developing a rule. This makes developing rules fast and easy. A number of other features of G2 used in ALPS are summarised by Walther (1992) and are available in the G2 user's manual.

2.4.4 Tuning rules

The tuning rules in the ALPS knowledge base were developed by Walther (1992) for a 2-input 2-output process. The focus of this study is to extend the applicability of the tuning rules to a process with up to 5-inputs and 5-outputs and to experimentally evaluate ALPS on a pilot-scale, computer-interfaced multivariable process. Certain additions were also made to the original knowledge base, to accommodate the on-line aspect of the experimental evaluation. An overview of the tuning rules and the tuning criteria are presented in the following sections.



2by2_plant generic

let the y1 of any 2by2 plant P -
the y of the mimo_output connected at the outlet-port-1 of P

let the y2 of any 2by2-plant P -
the y of the mimo_output connected at the outlet-port-2 of P

let the ysp1 of any 2by2-plant P -
the ysp of the mimo_output connected at the outlet-port-1 of P

let the ysp2 of any 2by2-plant P -
the ysp of the mimo_output connected at the outlet-port-2 of P

let the u1 of any 2by2-plant P -
the u of the mimo_input connected at the inlet port-1 of P

let the u2 of any 2by2-plant P -
the u of the mimo_input connected at the inlet port-2 of P

OUTPUT1, a mimo_output

Notes	OK
User restrictions	none
Names	OUTPUT1
Y	2.25
Y_ll	1.8
Y_hl	2.7
Outputweight	1.0
Ysp	2.25
Max_overshoot	0.3
Regulation_tolerance	0.2
Servo_status	0
Desired_tr	15.0
Observed_tr	11.157 *

INPUT1, a mimo_input

Notes	OK
User restrictions	none
Names	INPUT1
U	-0.196
Du	-5.69108e-7
U_ll	-1.0e6
U_hl	1.0e6
Du_ll	-1.0e6
Du_hl	1.0e6
Lambda	1.0e-6
Max_var_of_u	1.0e-2

Operator Logbook 3 Jun 91 ▼ ▲ Page 1

#78 3:56:20 p.m. Pause while running KB. You may resume, reset, or restart.

STRIPPER, a 2by2-plant

Notes	OK
User restrictions	none
Names	STRIPPER
N1	1
N2	15
Nu	1
Max_delay	2 *
Tau_over_ts	4.481
T_filter_a1	-0.8
T_filter_a2	0.0
U1	-0.196
U2	0.978
Y1	2.25
Ysp1	2.25
Y2	-2.04716e-7
Ysp2	0.0

Figure 2.5 The windowed environment of G2

2.4.4.1 Organization of tuning rules

The tuning rules are organised with a hierarchy as shown in figure 2.6. At the very top is the global supervisor, which contains a set of rules used to set initial tuning parameters and constraints, check the connection between G2 and the external multivariable GPC programs and provide an error message if the connection has failed. One of the nice features of G2 as mentioned before is object oriented programming. This allows the user to specify rules applicable to a class of objects with the same characteristics, in one statement, rather than have a separate rule for each object. An example of this is one of the global supervision rules. The rule reads as follows:

```
Unconditionally
set the y_hl of OUTP to 1.0e+06 and
set the y_ll of OUTP to -1.0e+06 and
set the u_hl of INP to 1.0e+06 and
set the u_ll of INP to -1.0e+06 and
set the du_hl of INP to 1.0e+06 and
set the du_ll of INP to -1.0e+06 and
set the outputweight of OUTP to 1.0 and
set the lambda of INP to 0.01 and
conclude that the servo_status of OUTP = 0
```

In this rule above INP and OUTP refer to the inputs and outputs of a multivariable system. Executing the above rule will set for example, the output weighting for all the outputs to 1.0. If different values of the output weighting need to be specified for different outputs then the rule for the output weighting which reads:

```
set the outputweight of OUTP to 1.0
```

has to be changed to read as follows:

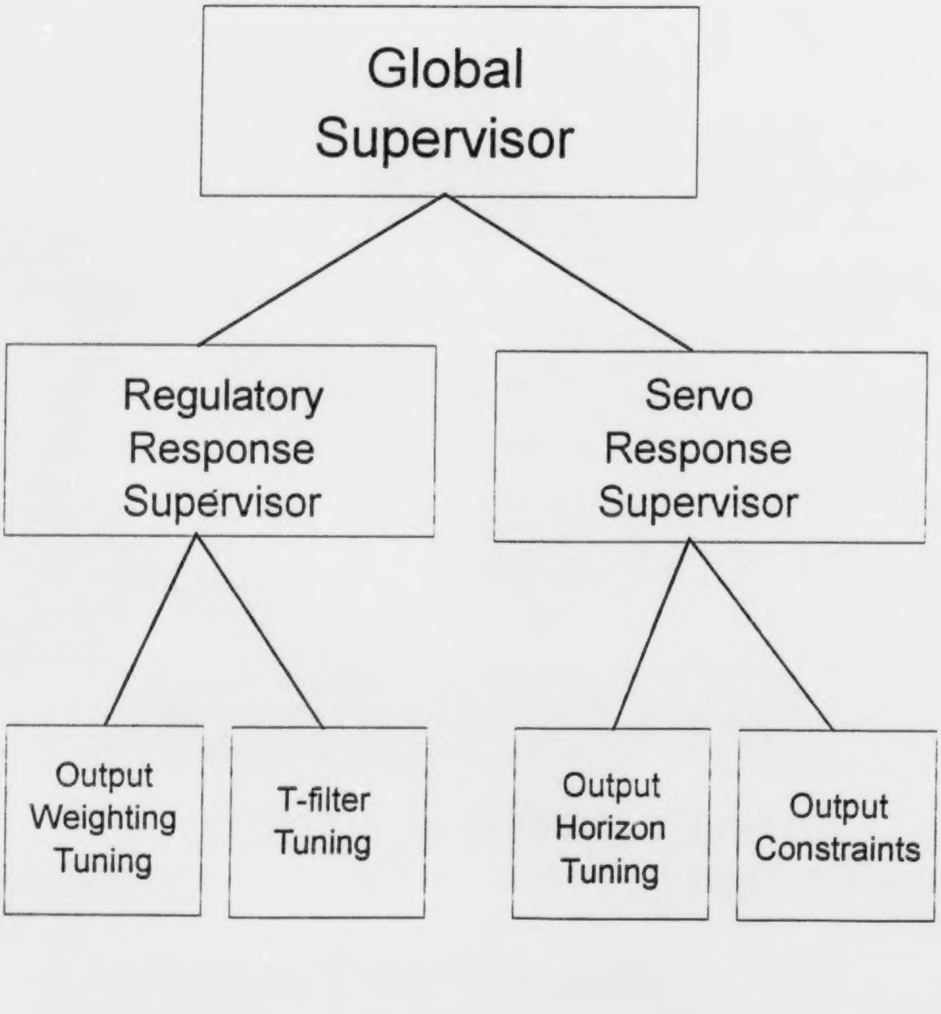


Figure 2.6 Organization of tuning rules

```
Set the outputweight of OUTPUT1 to 0.9 and
set the outputweight of OUTPUT2 to 0.8 and
set the outputweight of OUTPUT3 to 0.7 and
set the outputweight of OUTPUT4 to 0.6 and
set the outputweight of OUTPUT5 to 0.5
```

Where the variables (or objects) OUTPUT1, OUTPUT2, OUTPUT3, OUTPUT4 and OUTPUT5 have the same characteristics as the 'class' OUTP. This rule represents the 'non-generic' or exhaustive method of initialising each output weighting. Clearly, the generic method of initializing the output weighting is a lot more efficient. This method of defining 'classes' (*e.g.* OUTP) to represent a group of 'objects' (*e.g.* OUTPUT1, OUTPUT2, etc.) with the same characteristics, can thus be time saving, efficient and flexible when it comes to writing rules.

Below the global tuning supervisor are the regulatory response supervisor and the servo response supervisor. The regulatory response supervisor consists of two tuning strategies, T-filter tuning and output weighting tuning, while the servo response supervisor consists of output constraints and N2 tuning.

The knowledge base consists of a number of *procedures* as well. A *procedure* is analogous to a procedure in Pascal or 'C' or a subroutine in FORTRAN. The *procedures* in G2 are called from various rules and each procedure performs a specific task or a group of tasks related to the same subject. An example of an output weighting tuning rule which calls a *procedure* is as follows:

```
for any mimo_plant PLANT
if(for every mimo_output OUTP connected to PLANT
        (the servo_status of OUTP = 0))
then
start OUTPUT_WEIGHTING(PLANT) and inform the operator for the next 10
seconds that
"Regulation Supervisor:
```

```
Updating output weighting for [the name of PLANT]"
```

A few things to note from the above rule are:

- The idea of generic rules is evident from this rule as well. The first statement which is "for any mimo_plant PLANT" suggests that this rule can be used for any multivariable plant.
- The rules are very understandable and read like sentences, making the writing of rules easy. This sentence-like structure of rules also makes them easily understandable to a person who knows nothing about the knowledge base.
- The *procedure* called from this rule is called "OUTPUT_WEIGHTING".
- A message is displayed on the users screen in the message window which informs him or her that the output weighting of a certain plant is being updated. Since more than one plant can be operated with the same knowledge base, it becomes important to be able to differentiate between different plants by naming them.
- This rule is executed at periodic intervals, the length of which is set by the user. In G2 terminology, this is known as the *scan interval*. The *scan interval* for the above rule is set to 2 minutes.

Another method by which rules can get executed is illustrated by the following rule which is part of the servo response supervisor. This rule is a part of the N2 tuning method.

```
for any mimo_output OUTP
for any mimo_plant connected to OUTP
whenever the observed_tr of OUTP receives a value and when
(for every mimo_output OUT connected to the mimo_plant (the observed_tr
of OUT has a current value))
then
invoke N2-tuning rules for the mimo_plant
```

This rule is executed whenever the attribute "observed_tr" receives a value and certain other conditions are met. This rule does not have a *scan interval*. There are a number of other rules and procedures which illustrate the flexibility of G2 in developing a knowledge base. The interested reader is referred to the M.Sc. thesis by Walther (1992).

2.4.4.2 Tuning strategy

To reiterate, the GPC parameters tuned by ALPS are:

1. The T-filter parameters and automatic switching between a first order and a second order T-filter.
2. The output weighting, γ.
3. Constraints on the outputs are implemented and adjusted automatically.
4. The maximum prediction horizon, N2 is adjusted automatically as well.

In order to tune the above mentioned parameters, criteria for performance had to be selected, so that, based on that criteria, a tuning strategy could be devised.

1. T-filter tuning:

Tuning of the T-filter takes place at periodic fixed intervals. The interval is specified by the user. A parameter "Tau_over_ts" which is the ratio of the time constant τ and the sampling time T_s is used as the tuning criteria. Tau_over_ts is related to the discrete time poles of the first and second order T-filters by the following relation:

$$c_1 = e^{-\frac{T_s}{\tau}} \tag{2.19}$$

Where c_1 is the discrete time pole of the T-filter.

If bounds are placed on c_1 in equation 2.19 of the form:

$$0.5 \leq c_1 \leq 0.95 \tag{2.20}$$

then the bounds on τ/T_s can be calculated as:

$$\begin{gathered}-\frac{1}{\ln(0.5)} \le \frac{\tau}{T_s} \le -\frac{1}{\ln(0.95)} \\ 1.442 \le \frac{\tau}{T_s} \le 19.496\end{gathered} \tag{2.21}$$

The value of Tau_over_ts is calculated by using the information from the variances of the inputs. The equation used to calculate τ/T_s is:

$$\frac{\frac{\tau}{T_s}(t) - \frac{\tau}{T_s}(t-1)}{\frac{\tau}{T_s}(t-1)} = w\left[\frac{\sigma^2_{u,\max}(t) - \sigma^2_{u,\min}}{\sigma^2_{u,\min}}\right] \tag{2.22}$$

where,

$\sigma^2_{u,\max}(t)$ = The maximum over all input variances σ^2_{ui} , i = 1,2, ... m.

$\sigma^2_{u,\min}$ = The minimum over all *user specified* input variances σ^2_{ui} , i = 1,2, ... m. It is important to note that while the maximum variance is measured from the current state of the input variables, the minimum variance is read from a specification required before the execution of the knowledge base. The function of the minimum variance is to scale the maximum variance, so an appropriate τ/T_s can be calculated. The input variance specifications ($\sigma^2_{u,\min}$) are calculated by executing the simulation and estimating the respective values of variances for each input at steady state. The variable w is a relaxation factor and is set to 0.2. The value of w is chosen by trial and error. A large value of w implies aggressive tuning of the T-filter while a small value implies gradual tuning.

Once the value of τ/T_s is calculated by using equation 2.22, the T-filter parameters are updated by calculating c_1 in equation 2.19. A more practical implementation of the T-filter is given by the following equation:

$$\frac{1}{T(q^{-1})} = \left[\frac{1-c_1}{1-c_1q^{-1}}\right]^2 = \frac{1-a_1-a_2}{1-a_1q^{-1}-a_2q^{-2}} \tag{2.23}$$

where a_1 and a_2 are the T-filter parameters implemented in the knowledge base. If a_2 is set to 0 then equation 2.23 will represents a first order T-filter. Rules are written so that a first order T-filter, if set to the upper bound of $c_1 = 0.95$ for two consecutive tuning periods, will result in the switch to a default second order T-filter. If a second order T-filter is tuned to its lower bound of $c_1 = 0.5$ for two consecutive tuning periods, then the switch is made from a second order to a first order T-filter.

2. Output weighting tuning:

Tuning the output weighting is based on the standard deviation of the outputs. A maximum deviation is determined from the current state of the outputs and all the standard deviations are scaled by this maximum deviation. The output weighting is then the square of this scaled standard deviation. The output weighting is thus always less than or equal to 1. The equation representing the tuning method is:

$$\gamma_i = \left[\frac{\sigma_{yi} + \sigma_{ymax}}{max\_dev + \sigma_{ymax}}\right]^2 \qquad (2.24)$$

where,

$i = 1,2, \ldots n.$

σ_{yi} = The standard deviation over a specified window of time of the i^{th} output connected to the process.

max_dev = The maximum over all σ_{yi} values.

σ_{ymax} = User specified constant.

γ_i = The output weighting calculated.

When a lot of noise is present, *i.e.* when $\sigma_{yi} >> \sigma_{ymax}$,then equation 2.24 reduces to the simple relationship:

$$\gamma_i \approx \left[\frac{\sigma_{yi}}{max\_dev}\right]^2 \qquad (2.25)$$

In the other extreme, when little noise is present *i.e.* $\sigma_{yi} << \sigma_{ymax}$, then the relationship becomes:

$$\gamma_i \approx 1 \qquad (2.26)$$

This tuning method will assign maximum output weighting to outputs which have a high standard deviation, and scale the output weighting for the other channels accordingly. Increasing the output weighting on a certain channel is the equivalent of saying "pay more attention to this channel", thus the standard deviation of that channel should reduce to nominal levels.

3. Output constraints:

Output constraints are very useful in real situations where there are certain hardware limitations on the value an output variable can take. For multivariable systems, a set point change in one output could drive the other outputs away from their respective steady states, due to the presence of interactions. Although output constraints cannot eliminate the interactions present in a multivariable system, inclusion of output constraints in the problem formulation of GPC can result in a reduction of these interactions.

It should be noted that inclusion of tight constraints on outputs may not be of practical importance because interactions due to disturbances could push outputs past their constraints, resulting in their violation. The behavior of the closed loop in such a situation would depend on the characteristics of the solution searching algorithm, and its constraint violation handling capabilities. A constrained optimization algorithm which does not handle this violation appropriately, could cause the closed loop to go unstable.

Output constraints are implemented quite easily in the knowledge base by specifying a regulation band r_b around the operating value of the output. The value of r_b is user specified and is normally set to ±10%. In the following equations, $w_i(t)$ represents the setpoint of output i.

for $w_i(t) > 0$:

$$\begin{aligned} y_{max,i} &= w_i(t)\,(1+r_b) \\ y_{min,i} &= w_i(t)\,(1-r_b) \end{aligned} \tag{2.27}$$

for $w_i(t) < 0$:

$$\begin{aligned} y_{max,i} &= w_i(t)\,(1-r_b) \\ y_{min,i} &= w_i(t)\,(1+r_b) \end{aligned} \tag{2.28}$$

for $w_i(t) = 0$:

$$\begin{aligned} y_{\max,i} &= +r_b \\ y_{\min,i} &= -r_b \end{aligned} \qquad (2.29)$$

Rules for the above equations are generic and applicable for a process with n-outputs and m-inputs.

4.Output horizon tuning:

The purpose of tuning N2 is to have some control over the speed of response of the outputs. The theory of GPC suggests that for increasing the speed of response, the value of N2 can be decreased, while increasing the value of N2 results in a slower process response. This can be proved by looking at the objective function of GPC in equation 2.4, in which N2 appears as a parameter in the summation of the error between the set point and the predicted output. A larger value of N2 means that the error can be minimized in a greater number of time steps, thus the control action calculated will be less aggressive and the output response slower. The opposite is true when N2 is made small. It should be noted that changing the value of N2 affects the closed loop process response of *all* the outputs for a multivariable system. For this tuning strategy to be affective, NU should be set to 1. Walther (1992) has shown that tuning N2 effects the speed of the closed loop response of a multivariable process only when NU, the control horizon, is set to 1. If NU is greater than 1, then the speed of the closed loop response of the process is not affected by changing N2.

The criteria for tuning N2 is the *rise time* of the outputs. The *rise time* is the time it takes for an output to reach 80% of its steady state value (or set point). The value of the *rise time* is calculated when a set point change is made in any of the outputs. The *rise time* calculated includes the time due to delays. For example, if a set point change is made from 1.0 to 2.0, then the *rise time* is the total time taken by the output (including the time due to delays) to reach a value of 1.6 (80% of 2.0 is 1.6) or greater. This "observed" value of the rise time ($\Delta t_{r,i}$ i=1,2, .. n) is compared with a user specified "desired" value of the rise time ($\Delta t_{rm,i}$ i=1,2, .. n) and a difference between the two rise times, $\Delta t_{r,i}$, is calculated. A maximum of all these $\Delta t_{r,i}$'s is determined, and this value, Δt_{r}, is used in further calculations. Based on simulations done by Walther, a correlation between Δt_r and $\Delta N2$ (the change in N2) is developed, and tuning is carried out by a set of rules which determine the value of Δt_r and implement a relevant $\Delta N2$. Once the value of $\Delta N2$ is determined, N2 is updated by the following equation:

$$N2(t) = N2(t-1) + \Delta N2 \tag{2.30}$$

with $N2_{min} \leq N2 \leq 40$, where

$$N2_{min} - NU \geq d_{max} - 1 \tag{2.31}$$

d_{max} = the maximum delay in the multivariable system.

The minimum value of N2, $N2_{min}$ is chosen based on the inequality 2.31 (Shah *et al.*, 1987), while the maximum value of N2 is set to 40. Correlations between Δt_r and $\Delta N2$ are in table 2.2. Although different values of N2 can be implemented for each output, for the purpose of simplicity, the same N2 is used for all outputs in this work.

Δt_r	$\Delta N2$ for $\Delta t_r < 0$	$\Delta N2$ for $\Delta t_r > 0$
$\lvert\Delta t_r\rvert < 1.0$	0	0
$1.0 \leq \lvert\Delta t_r\rvert < 2.0$	-1	1
$2.0 \leq \lvert\Delta t_r\rvert < 4.0$	-2	2
$\lvert\Delta t_r\rvert \geq 4.0$	-5	5

Table 2.2 Correlation among Δt_r and $\Delta N2$

2.4.4.3 Modifications to the knowledge base

The ALPS knowledge base discussed in the preceding sections was designed by Walther (1992) for a 2-input 2-output multivariable system. Modifications to this knowledge base had to be made by the author to accommodate the 5-input, 5-output multivariable system application considered in this work.

The most significant change made to the ALPS knowledge base is to devise a method to avoid an exhaustive procedure to determine if a *rise time* calculation can be performed on a specific output. The following issues are important when calculating the *rise time* for an output.

- All outputs other than the one for which the *rise time* calculation is requested, should be at steady state, thus requiring a method to automatically detect this state. This is to avoid interactions from other outputs showing up in the output for which the *rise time* calculation is being carried out.
- During the calculation of the *rise time*, no set point changes should be allowed in other outputs.
- One 'attribute' of the 'class' OUTP (standing for output) is 'servo_status'. This attribute is incremented every time a set point change is made. A suitable method to decrement the 'servo_status' once steady state is reached is also implemented.

For a 2-input 2-output process, the exhaustive method used by Walther to check the 'servo_status' for a feasible *rise time* calculation results in 2 rules (${}^{2}P_{2} = 2! / (2-2)! = 2! = 2*1 = 2$). However, for a 5-input, 5-output process, using the same method would result in 120 rules (${}^{5}P_{5} = 5! / (5-5)! = 5! = 5*4*3*2*1 = 120$). This is clearly very tedious and inefficient. After some thought, it was found that only 5 different rules were required for the 5-input 5-output problem, along with modifications to the *rise time* calculation procedure. A listing of the pertinent rules and the modified versions are presented in Appendix A.

ALPS also consists of generic tuning rules which can be applied to a multivariable process with any number of inputs and outputs. However, the user interface of ALPS, consisting of graphs and other displays is designed for a multivariable system with 2 inputs and 2 outputs. Thus, the user interface had to be expanded to accommodate the 5- input, 5-output multivariable system application. The GSI program, which ensures a one to one correspondence between the variables in the knowledge base and the variables used in the external control algorithms, is also designed for a 2-input, 2-output multivariable system and had to be expanded for the 5-input 5-output system implemented by the author.

A manual control panel was added on to the ALPS knowledge base so that open loop simulations could be performed and the experimental pilot plant could be operated in manual control mode. After operating the pilot plant in manual mode, the switch to automatic mode is made with a "bumpless transfer" *i.e.* there is no deviation in the outputs when the transfer from manual mode to automatic mode or *vice versa*, is made.

2.5 Summary

This chapter focusses on two aspects, the Generalized Predictive Control Algorithm (GPC) and the Adaptive Long range predictive control Process Supervisor (ALPS).

The GPC algorithm belongs to the family of Long Range Predictive Control (LRPC) algorithms which have the feature of minimizing the errors between the set point and the predicted output over a user specified time frame. Expressing the sum of the errors over the user specified time frame results in an equation known as the objective function. Minimizing this objective function with respect to the change in the control action, Δu, results in an equation which is used to calculate the control action at every time step. Constraints on the absolute magnitude of the control action, the rate of change of the control action and the absolute magnitude of the output can be incorporated in the problem formulation, and the resulting constrained optimization problem is solved numerically by a commercial quadratic programming solution package called QPSOL.

The ALPS knowledge base developed using Gensym's G2 development system is used to tune various parameters of GPC in order to improve the characteristics of the closed loop response. ALPS is used to tune the output weighting, T-filter, maximum prediction horizon (N2) and to enable output constraints. The criteria for tuning the output weighting is related to the standard deviation of the outputs, the criteria for tuning the T-filter is based on the variance of the inputs while tuning of N2 depends on the rise time of the outputs when a set point change is made.

All the tuning rules included in ALPS are generic and can be applied to any multivariable plant with n outputs and m inputs. Significant changes in the procedure to calculate the *rise time* were made in order to avoid the exhaustive method used previously. Unfortunately, version 2.0 of G2 does not support variable dimensions in its GSI (G2 Systems Interface) program, thus the GSI program, designed for a 2-input 2-output process in ALPS, was modified to accommodate the 5-input 5-output problem. The newer version (3.0) of G2 supports arrays and allows for greater flexibility in variable dimension programming. A manual control panel is added on to ALPS to facilitate the experimental evaluation of ALPS.

Chapter 3

Issues in multivariable control

There are a number of reasons why multivariable processes are difficult to control. A systematic study of the characteristics of such processes is necessary in control systems design. The purpose of this chapter is to bring forward some issues which are important in understanding multivariable process dynamics. This chapter is organized as follows: A review of previous work on multivariable control is presented in section 3.1. The review focusses on implementations of multivariable control strategies on large dimensional processes. The structure and complexity of multivariable systems is discussed in section 3.2, followed by stability of multivariable systems in section 3.3. Finally, conclusions are presented in section 3.4.

3.1 Review of previous work on multivariable control

This review outlines a few multivariable control techniques. In particular, multi-loop PID, state-space and a few model based techniques such as DMC, MOCCA and GPC are considered here. Some applications of DMC on large industrial processes are reviewed so as to compare their complexity with the MIMO GPC application considered in this work.

Muti-loop PID is the most popular control strategy in industry due to the simple mathematical formulation of the controller and the ease with which a control loop can be configured. Most muti-loop PID controllers are tuned by treating each loop as if it were independent. This, however, results in a number of problems such as the question of how to pair loops, iterative and difficult tuning of loops and instability caused by interactions, even if each transfer function is stable. If a lot of interaction is present in the process, then the PID loops are usually 'detuned' or other loop configurations such as cascade loops are used. If the process does not have a lot of interactions then the transfer function matrix is made diagonally dominant so that loop pairing can become straightforward and tuning becomes easier. A number of methods such as Bristol's relative gain array (RGA), Nyquist array and characteristic locus methods are used for designing multivariable controllers.

Recent work on multi-loop controllers is Individual Channel Design or ICD technique of O'Reilly and Leithead (1991). ICD consists of simplifying the block diagram representation of a 2-input 2-output process and then using certain assumptions to obtain controllers for the system. From their methodology, it is clear that using the ICD technique on a larger system of say, 5 inputs and 5 outputs would be algebraically demanding. A subsequent paper by Leithead and O'Reilly (1991) looks at the effect of right half plane zeroes and right half plane poles in the process transfer function on the performance of a closed loop. A set of conditions were then derived for a controller to be stabilising. These conditions result in a controller which performs pole-zero cancellation. In reality pole-zero cancellation may not be exact and can result in instability, thus being a limitation of this design. Another limitation of the ICD technique is that it cannot handle delays in process transfer functions. Since many chemical processes contain delays, this is a serious limitation.

A different method of representing a process is the state-space representation. This representation gives an *internal* description of a process, obtained directly from linearizing ordinary differential equation (ODE) representation of the process. A unique transfer function representation can be obtained from this state-space description by simply taking its Laplace transform. A unique state-space description, however, cannot be obtained from a transfer function description. The idea behind state feedback control is to modify a given state-space system so as to obtain a new closed loop system with specific poles (or eigenvalues). A drawback of the state-space approach is that sometimes a detailed *internal* description of the process is either not available or too complicated, resulting in the failure of this technique.

There are a number of model based control techniques such as the Dynamic Matrix Control or DMC algorithm (Cutler and Ramaker, 1979), Mutivariable, Optimal, Constrained Control Algorithm or MOCCA (Sripada and Fisher, 1985) and the Generalized Predictive Control or GPC algorithm (Clarke *et al.*, 1987). All these control techniques need a process model description either in step response coefficients or transfer function form. All these techniques can also incorporate constraints in their problem formulation, thus are of practical use to industry.

There are a number of applications of DMC. Cutler and Hawkins (1987) applied the DMC algorithm to a 7-input 4-output hydrocracker reactor. A subsequent application of DMC was also made on the second stage of the hydrocracker (Cutler and Hawkins, 1988). This subsequent application had 4 inputs and 4 outputs. In both these applications, it is noticeable that the dynamic matrix used by the control algorithm is diagonally dominant, reducing interactions among channels and making the process easier to control. In contrast, the process considered in this work has 5 inputs and 5 outputs, is highly interactive and is not diagonally dominant.

Other more complex implementations of DMC on large MIMO systems are as follows: Park (1988) implemented DMC on a catalytic cracking unit with 8 inputs and 8 outputs and a number of constraints. Hokanson *et al.* (1989) applied DMC to a refrigerated fractionator with 7 inputs and 4 outputs. Tran and Cutler (1989) applied DMC to benzene and toluene towers with 3 inputs, 6 outputs and 3 inputs, 4 outputs respectively. Cutler and Finlayson (1988), Glova *et al.* (1989), VanHoof *et al.* (1989) have all applied DMC to "large" multivariable processes with a number of inputs and output. In each case economic benefits were generated and process problems were solved. Since DMC and GPC are both long range predictive control strategies, GPC can be used with some confidence on "large" multivariable problems as well (such as the 5-input 5-output problem in this work).

Recent applications of GPC on experimental equipment are by Mutha

(1990) on a 2-input 2-output stirred tank system and Dion *et al.* (1991) to control an environment test chamber. A modified form of the original GPC algorithm was used by Dion *et al.*, with an additional parameter (the value of the control action, u) in the cost function, J. An additional weighting parameter on the value of the control action is used as well. In both applications, the performance achieved by using GPC and the modified form of GPC is much better than that achieved previously by using PID loops.

A good summary of the benefits of real time optimisation with multivariable control is in the paper by Cutler and Perry (1982). They conclude that on-line optimisation and control can amount to 6 - 10% of the added value of a given process. Additional benefits of real-time optimization include an enhanced understanding of the process and distributing this know-how within the organization.

Thus a multivariable control strategy can be a powerful tool in the solution of process control problems.

3.2 Structure and complexity of multivariable systems

A compact way of representing a multivariable system in transfer function matrix form is:

$$\underline{y} = \underline{\underline{G}}\,\underline{u} \tag{3.1}$$

Where y is a vector of outputs, u is a vector of inputs and G is a matrix of transfer functions. For a process which has 3 inputs and 3 outputs, equation 3.1 can be expanded as follows:

$$\begin{bmatrix} y1 \\ y2 \\ y3 \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} & G_{13} \\ G_{21} & G_{22} & G_{23} \\ G_{31} & G_{32} & G_{33} \end{bmatrix} \begin{bmatrix} u1 \\ u2 \\ u3 \end{bmatrix} \tag{3.2}$$

The output y1 is thus a product of transfer functions G_{11}, G_{12}, G_{13} and inputs u1, u2, u3 respectively. This can be written as:

The complexity of the process is determined by the structure and contents of the transfer

$$y1 = G_{11}*u1 + G_{12}*u2 + G_{13}*u3 \tag{3.3}$$

function matrix $\underline{G}$. Typical structures of the $\underline{G}$ matrix are square, rectangular, diagonally dominant and diagonal. A square $\underline{G}$ matrix appears in processes which have the same number of inputs and outputs. Another characteristic of such a system is that each output (y) is affected by each input (u). An example of such a process is given by equation 3.2. A rectangular $\underline{G}$ matrix occurs in processes which have a non-square structure *i.e.* the number of inputs does not equal the number of outputs. One such example can be of a process with 3 outputs and only 2 inputs. This can be represented by equation 3.4. Such a system is harder to control because it is *underspecified*, that is, the number of inputs is less than the number of outputs. An *overspecified* system is when there are more inputs than outputs.

$$\begin{bmatrix} y1 \\ y2 \\ y3 \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \\ G_{31} & G_{32} \end{bmatrix} \begin{bmatrix} u1 \\ u2 \end{bmatrix} \tag{3.4}$$

The idea of diagonal and diagonally dominant applies to square $\underline{G}$ matrices only. Consider the 3-input 3-output system in equation 3.2. The system would be diagonal if the transfer functions G_{12}, G_{13}, G_{21}, G_{23}, G_{31} and G_{32} were all zero. In such a situation, the outputs y1, y2 and y3 would be given by:

$$\begin{aligned} y1 &= G_{11}*u1 \\ y2 &= G_{22}*u2 \\ y3 &= G_{33}*u3 \end{aligned} \tag{3.5}$$

This system is the easiest to control because each output can be "paired" with a single input. Also each input affects only one output, unlike the fully interactive system represented by equation 3.3 in which each input affects each output. The idea of a diagonally dominant system is clearer in the case of large multivariable systems. Consider a 6-input 6-output system given by equation 3.6.

There are a number of formal mathematical definitions of diagonal dominance, however, in this work a system is considered diagonally dominant if either of the following is true:

- The gains and time constants of transfer functions on the diagonal (G_{11} to G_{66}) are relatively larger than the gains and time constants of other off-diagonal transfer function elements.
- Most of the off-diagonal transfer functions are zero.

$$\begin{bmatrix} y1 \\ y2 \\ y3 \\ y4 \\ y5 \\ y6 \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} & G_{13} & G_{14} & G_{15} & G_{16} \\ G_{21} & G_{22} & G_{23} & G_{24} & G_{25} & G_{26} \\ G_{31} & G_{32} & G_{33} & G_{34} & G_{35} & G_{36} \\ G_{41} & G_{42} & G_{43} & G_{44} & G_{45} & G_{46} \\ G_{51} & G_{52} & G_{53} & G_{54} & G_{55} & G_{56} \\ G_{61} & G_{62} & G_{63} & G_{64} & G_{65} & G_{66} \end{bmatrix} \begin{bmatrix} u1 \\ u2 \\ u3 \\ u4 \\ u5 \\ u6 \end{bmatrix} \tag{3.6}$$

Such a system would also be relatively easy to control because of few interactions among different outputs.

3.3 Stability of multivariable systems

There are a number of parameters such as sampling time, which affect the stability of a multivariable system. It is important to understand the affect of varying these parameters on the closed loop and open loop stability of a multivariable system, so that an appropriate parameter value can be chosen when designing a controller. In this work, the transfer functions in the transfer function matrix $\underline{G}$, are all open loop stable, meaning that the instability is not present inherently, but is as a result of subsequent choice of certain design parameters.

3.3.1 Multiple time scale systems[1]

The idea of multiple time scale systems is very apparent when there are a number of transfer functions present in the $\underline{G}$ matrix and each transfer function has a different time constant. This can be best illustrated by an example. Consider a process described by the transfer function matrix in equation 3.7.

---

[1] The simulations were performed by discretizing the transfer function matrix in cascade with a zero order hold device using MATLAB functions

$$\underline{\underline{G}} = \begin{bmatrix} \frac{4.05e^{-27s}}{50s+1} & \frac{1.77e^{-28s}}{60s+1} & \frac{5.88e^{-27s}}{50s+1} & \frac{1.20e^{-27s}}{45s+1} & \frac{1.44e^{-27s}}{40s+1} \\ \frac{5.39e^{-18s}}{50s+1} & \frac{5.72e^{-14s}}{60s+1} & \frac{6.90e^{-15s}}{40s+1} & \frac{1.52e^{-15s}}{25s+1} & \frac{1.83e^{-15s}}{20s+1} \\ \frac{3.66e^{-2s}}{9s+1} & \frac{1.65e^{-20s}}{30s+1} & \frac{5.53e^{-2s}}{40s+1} & \frac{1.16}{11s+1} & \frac{1.27}{6s+1} \\ \frac{5.92e^{-11s}}{12s+1} & \frac{2.54e^{-12s}}{27s+1} & \frac{8.10e^{-2s}}{20s+1} & \frac{1.73}{5s+1} & \frac{1.73}{19s+1} \\ \frac{4.13e^{-5s}}{8s+1} & \frac{2.38e^{-7s}}{19s+1} & \frac{6.23e^{-2s}}{10s+1} & \frac{1.31}{2s+1} & \frac{1.26}{22s+1} \end{bmatrix} \qquad (3.7)$$

From this process, it is evident that the time constants differ widely. The time constants vary from as high as 60 seconds to as low as 2 seconds. Since, for the purposes of discrete controller design, this system has to be discretised, *i.e.* an equivalent representation has to be determined in the discrete domain, choosing a sampling time becomes a challenging problem. Thus transfer function $\underline{\underline{G}}$ in equation 3.7 has *multiple time scales*. In order to demonstrate the sensitivity of this system to the sampling time, consider simulations with sampling time as 2 seconds and sampling time 5 seconds. The sampling time of 5 seconds is chosen based on a *rule* which can be followed for obtaining a rough estimate of the sampling time: *Choose the sampling time such that it is 1/10th the dominant time constant of the process.* The dominant time constant from equation 3.7 is 60 seconds, thus 5 seconds, which is approximately 1/10th of 60, is chosen as the sampling time. The sampling time of 2 seconds is chosen to contrast the simulation results with those obtained with the sampling time of 5 seconds. These simulations are presented in figures 3.1 to 3.4. The simulation presented in both cases are closed loop simulations for a set point change in y1. The reader is referred to section 2.2.2 for a description of the following GPC tuning parameters used in simulations.

N1 = 5; N2 = 15; NU = 2

$\gamma_{1,2,3,4,5} = 1.0$

$\lambda_{1,2,3,4,5} = 0.01$

$U_{max,min(1,2,3)} = \pm 0.5$; $U_{max,min(4,5)} = \pm 10.0$

$\Delta U_{max,min} = \pm 0.05$

T-filter = $1 - 0.8q^{-1}$ (1st order T-filter)

The following observations can be made from figures 3.1 and 3.2:

- The outputs y1 to y5 in figure 3.1 do not track their respective set points. The output y3 specifically starts to diverge after sample time 100. The inputs u1 to u5 in figure 3.2 also do not reach a steady state and diverge slowly.

In order to contrast the above closed loop performance, another simulation with sampling time as 5 seconds was performed. The results from this simulation are presented in figures 3.3 and 3.4. From this simulation we can say the following:

- All the outputs reach steady state by sample time 200 and track their respective set points.

Thus the sampling time seems to play a very important role in the closed loop response of the MIMO process (equation 3.7). This simulation is not typical of all MIMO systems but represents the observations made by the author with the system represented by equation 3.7.

3.3.2 Poorly conditioned systems

A system is poorly conditioned when two columns of the Hessian matrix are almost equal. The Hessian matrix is given by $(G^TG + \lambda I)$, where G is the step response matrix, λ is a weighting factor chosen by the user and I is the identity matrix containing 1's on the diagonal. The step response matrix 'G' should not be confused with '$\underline{G}$' which is the transfer function matrix (equation 3.7). If the system is poorly conditioned, then the inverse of the Hessian matrix, will not be accurate and will result in erroneous calculation of the control action, which may result in diverging outputs or offset at steady state. In order to detect if the system is poorly conditioned, it is necessary to calculate the condition number of the Hessian matrix. A library function is written in 'C' to take care of the calculation of the condition number. If the condition number is greater than 10^5, then the Hessian matrix can be considered as poorly conditioned, which is undesirable.

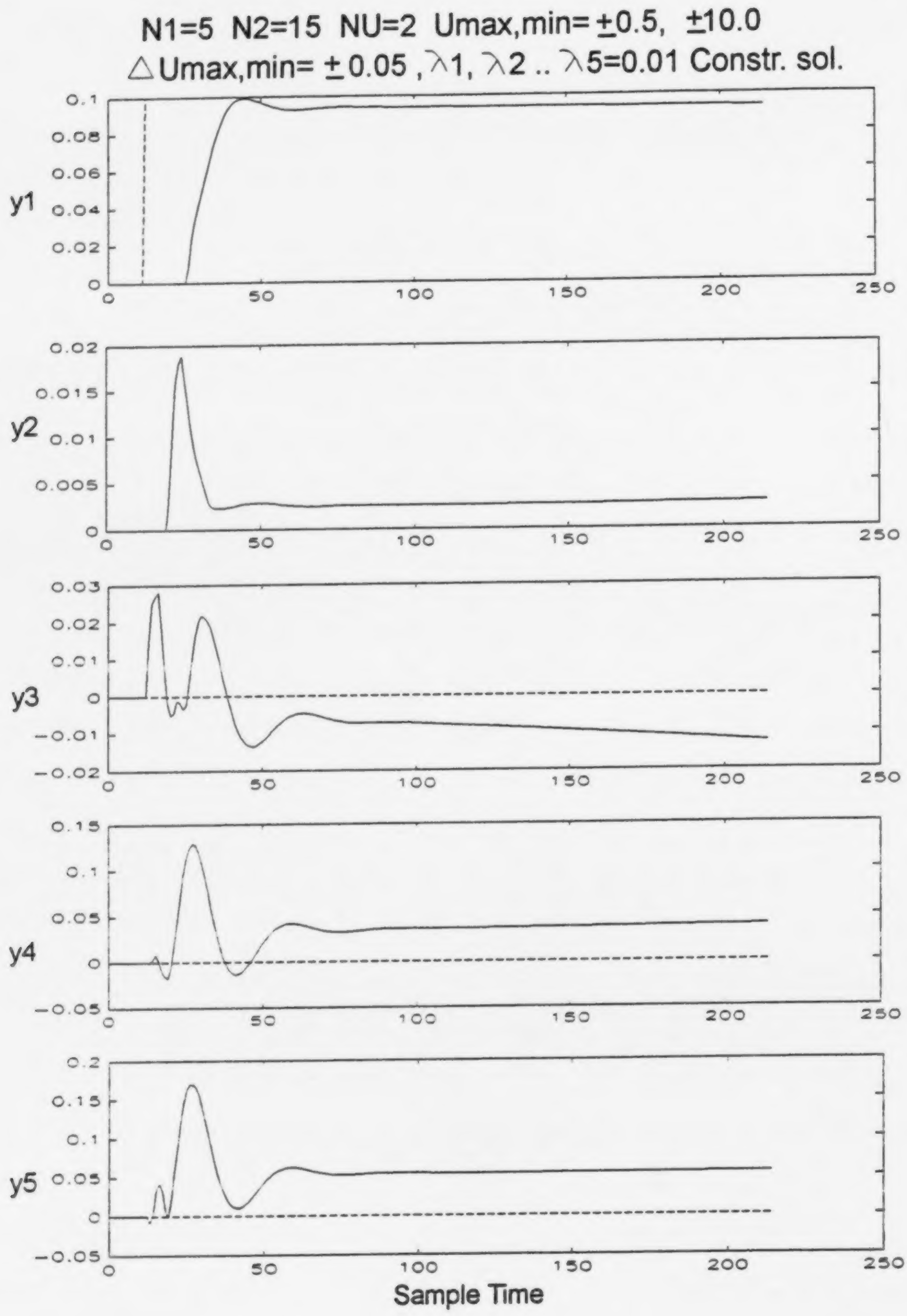


Figure 3.1 Simulation of MIMO system (3.7) with sampling time of 2 seconds

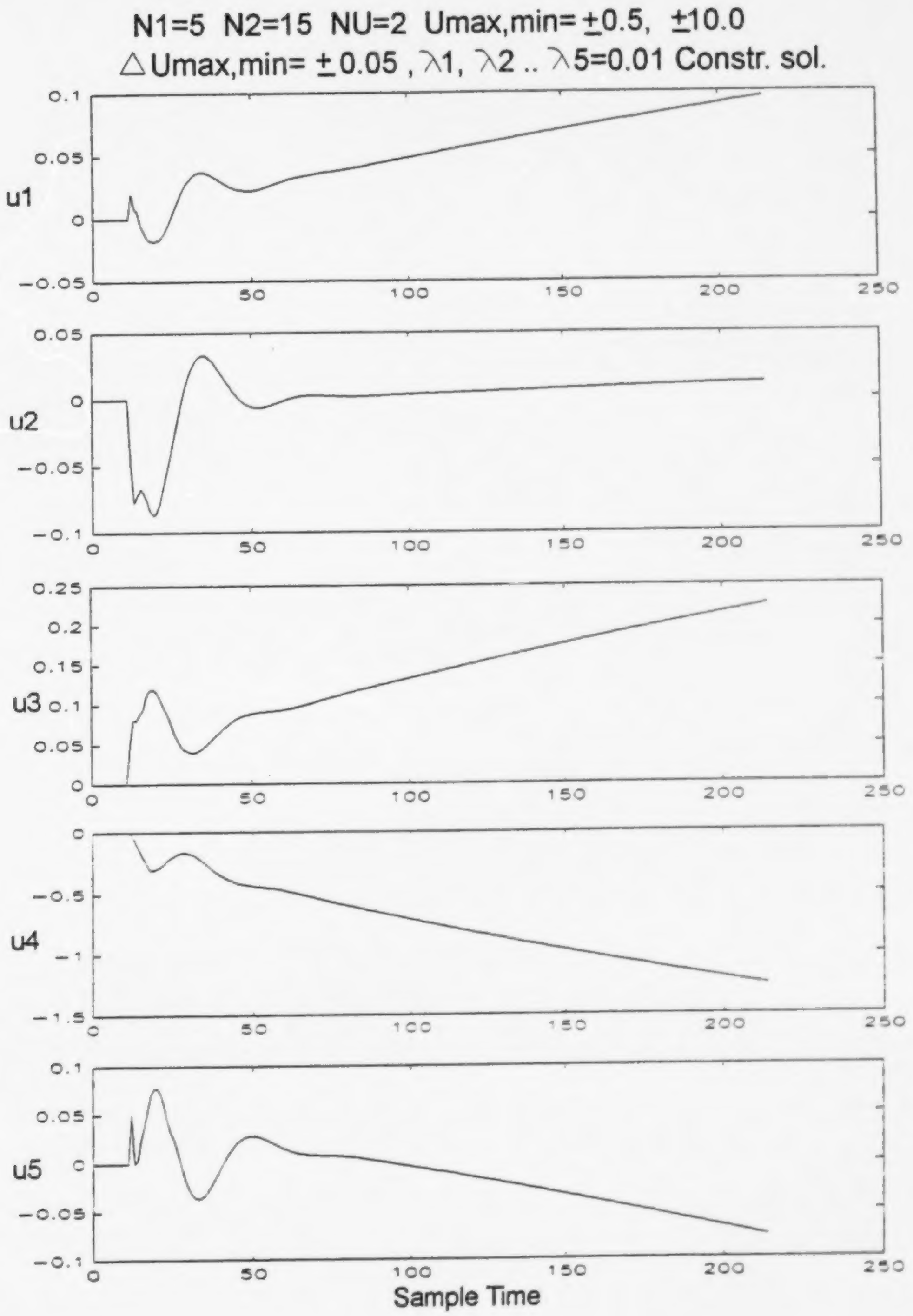


Figure 3.2 Simulation of MIMO system (3.7) with sampling time of 2 seconds

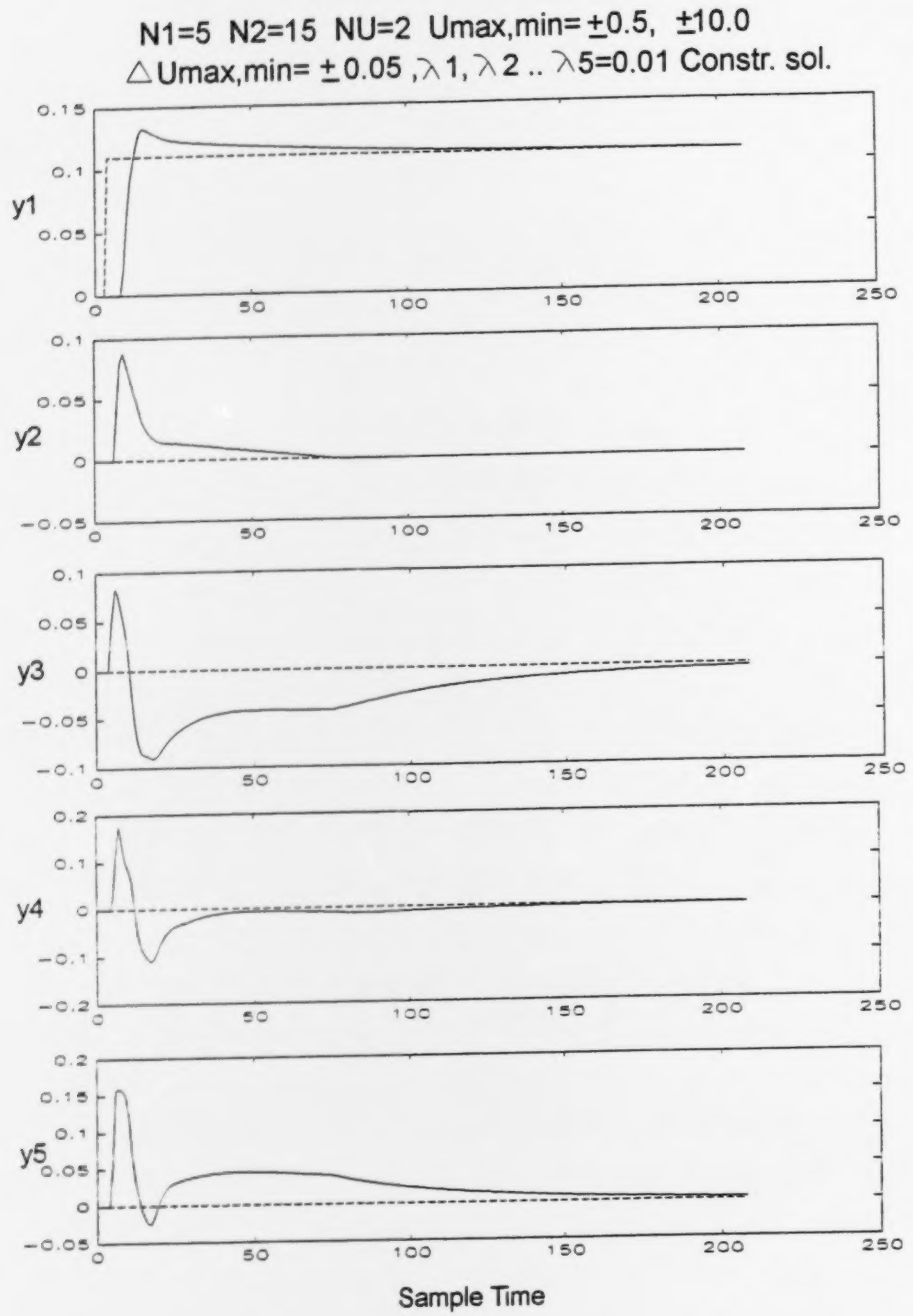


Figure 3.3 Simulation of MIMO system (3.7) with sampling time of 5 seconds

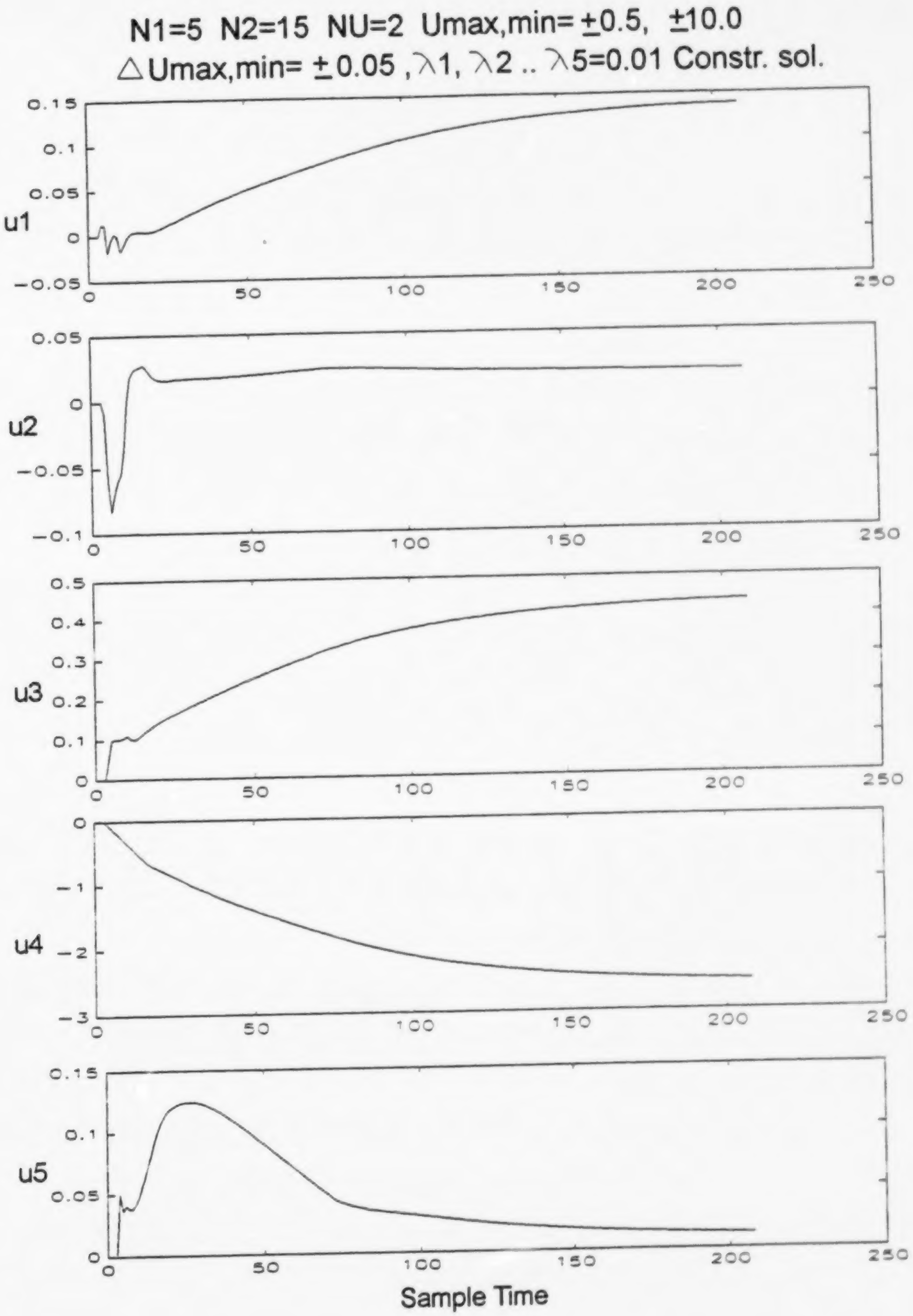


Figure 3.4 Simulation of MIMO system (3.7) with sampling time of 5 seconds

One method of making the Hessian well-conditioned is by increasing the output weighting, λ.

As an example, consider the transfer function matrix $\underline{G}$ in equation 3.7. If two columns of $\underline{G}$ are made identically equal, the step response matrix G will contain at least two columns which are identical, resulting in a poorly conditioned Hessian. The modified $\underline{G}$ matrix with column 1 and column 3 identical is as follows:

$$\underline{G} = \begin{bmatrix} \frac{4.05e^{-27s}}{50s+1} & \frac{1.77e^{-28s}}{60s+1} & \frac{4.05e^{-27s}}{50s+1} & \frac{1.20e^{-27s}}{45s+1} & \frac{1.44e^{-27s}}{40s+1} \\ \frac{5.39e^{-18s}}{50s+1} & \frac{5.72e^{-14s}}{60s+1} & \frac{5.39e^{-18s}}{50s+1} & \frac{1.52e^{-15s}}{25s+1} & \frac{1.83e^{-15s}}{20s+1} \\ \frac{3.66e^{-2s}}{9s+1} & \frac{1.65e^{-20s}}{30s+1} & \frac{3.66e^{-2s}}{9s+1} & \frac{1.16}{11s+1} & \frac{1.27}{6s+1} \\ \frac{5.92e^{-11s}}{12s+1} & \frac{2.54e^{-12s}}{27s+1} & \frac{5.92e^{-11s}}{12s+1} & \frac{1.73}{5s+1} & \frac{1.73}{19s+1} \\ \frac{4.13e^{-5s}}{8s+1} & \frac{2.38e^{-7s}}{19s+1} & \frac{4.13e^{-5s}}{8s+1} & \frac{1.31}{2s+1} & \frac{1.26}{22s+1} \end{bmatrix} \qquad (3.8)$$

The system represented by equation 3.8 is simulated with λ equal to 0.0 and λ equal to 1.0. The output weighting is set to 1.0 for all the outputs ($\gamma_{1,2,3,4,5} = 1.0$). Results are in figures 3.5 to 3.8. The following can be said about these simulations:

- When the λ weighting is set to 0.0 (figures 3.5 and 3.6), the outputs show an oscillatory response. Another interesting observation is that none of the inputs saturate at their amplitude constraint, indicating that the set point in y1 of magnitude 0.05 is a feasible set point.
- Increasing the λ weighting to 1.0 (figures 3.7 and 3.8), results in marked improvement in the response, with no more oscillations appearing in the outputs. However, the outputs drift and do not track their respective set points.

In order to eliminate drift in the outputs, another simulation is performed with λ weighting increased to 20.0. The output weighting is also changed from 1.0 on all outputs to 1.0 on output 1 and 0.5 on all other outputs ($\gamma_1 = 1.0$, $\gamma_{2,3,4,5} = 0.5$). The selection of this output weighting is based on the knowledge that a set point change is requested in

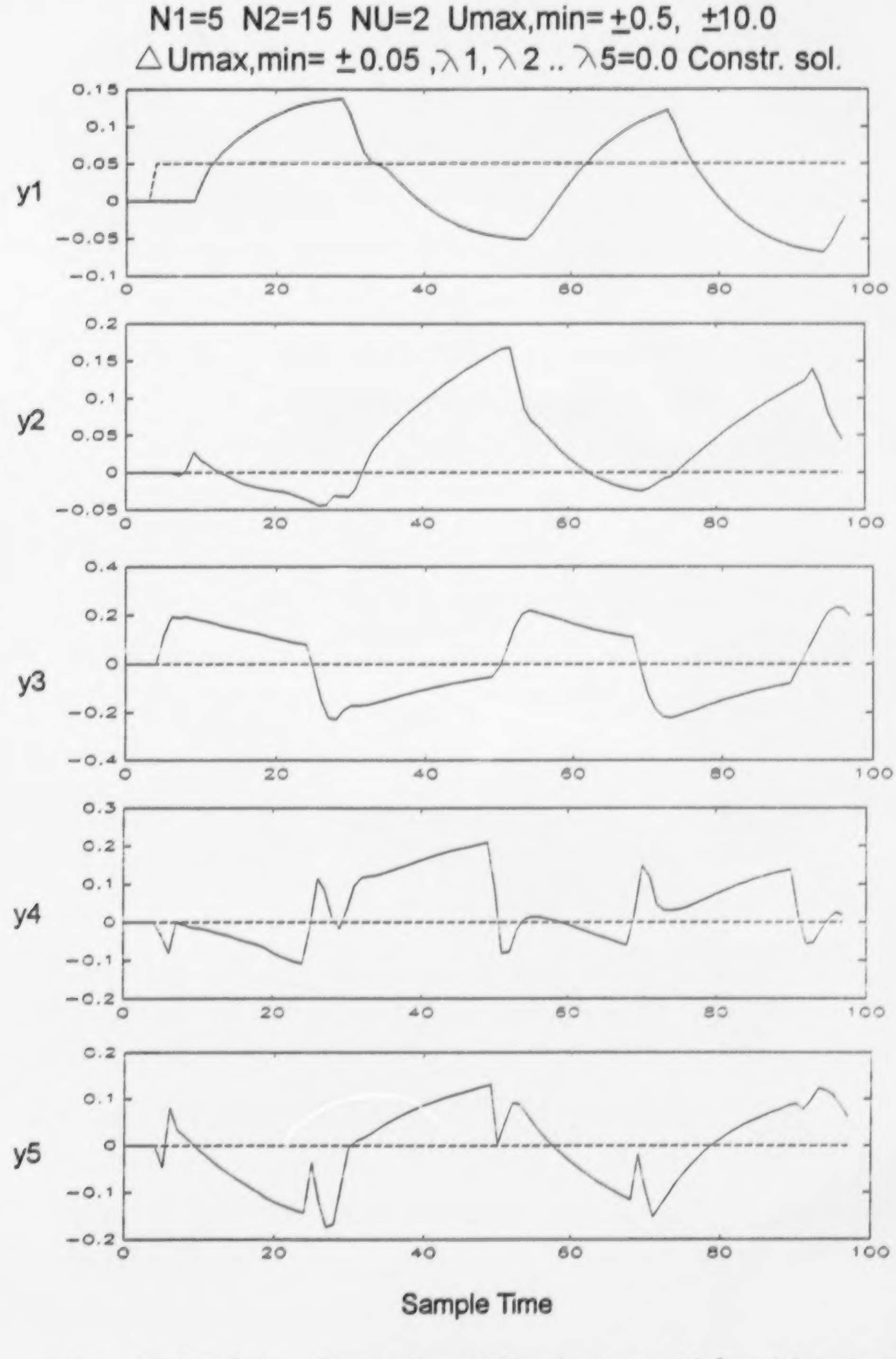


Figure 3.5 Simulation of a poorly conditioned system with λ weighting set to 0.0

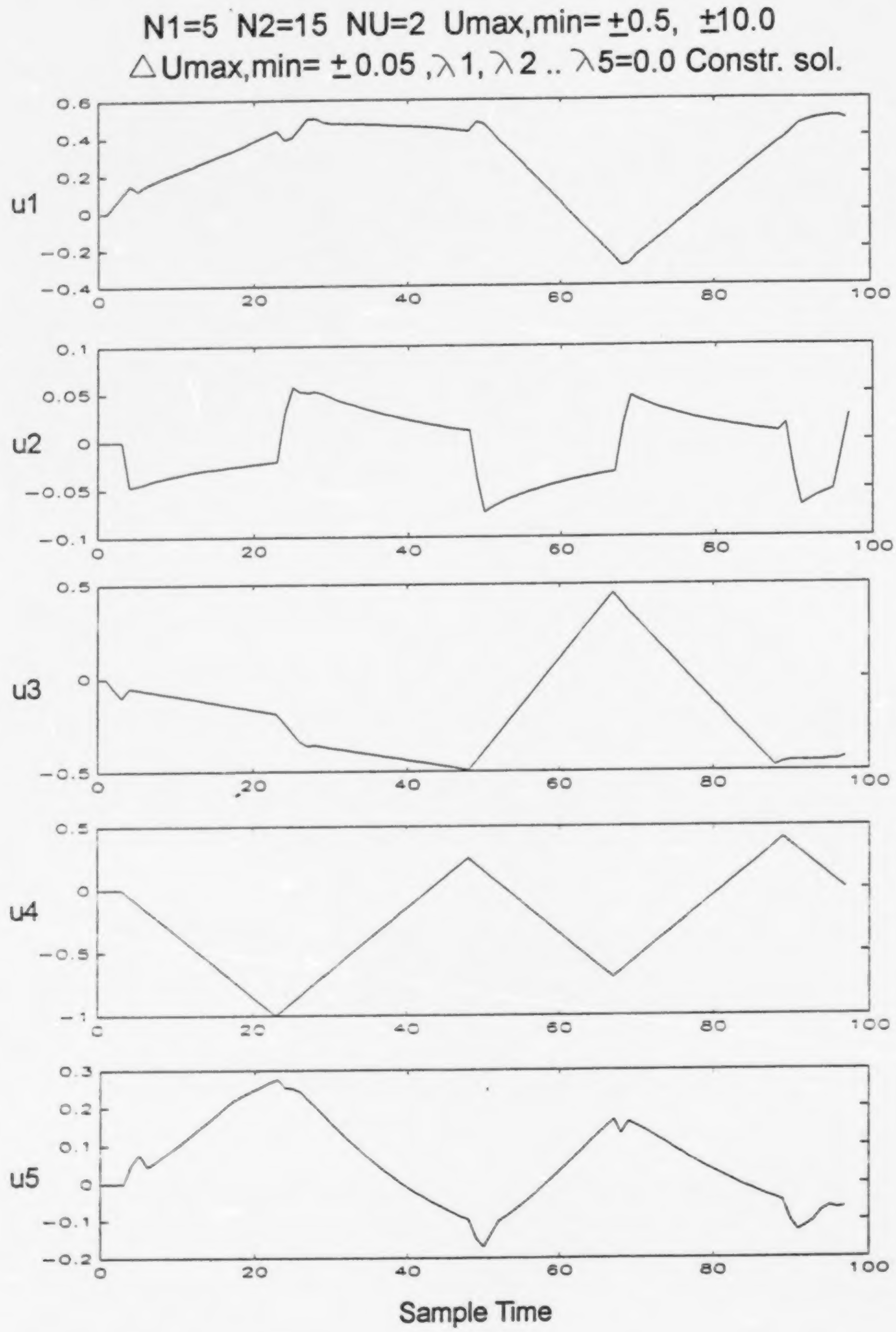


Figure 3.6 Simulation of a poorly conditioned system with λ weighting set to 0.0

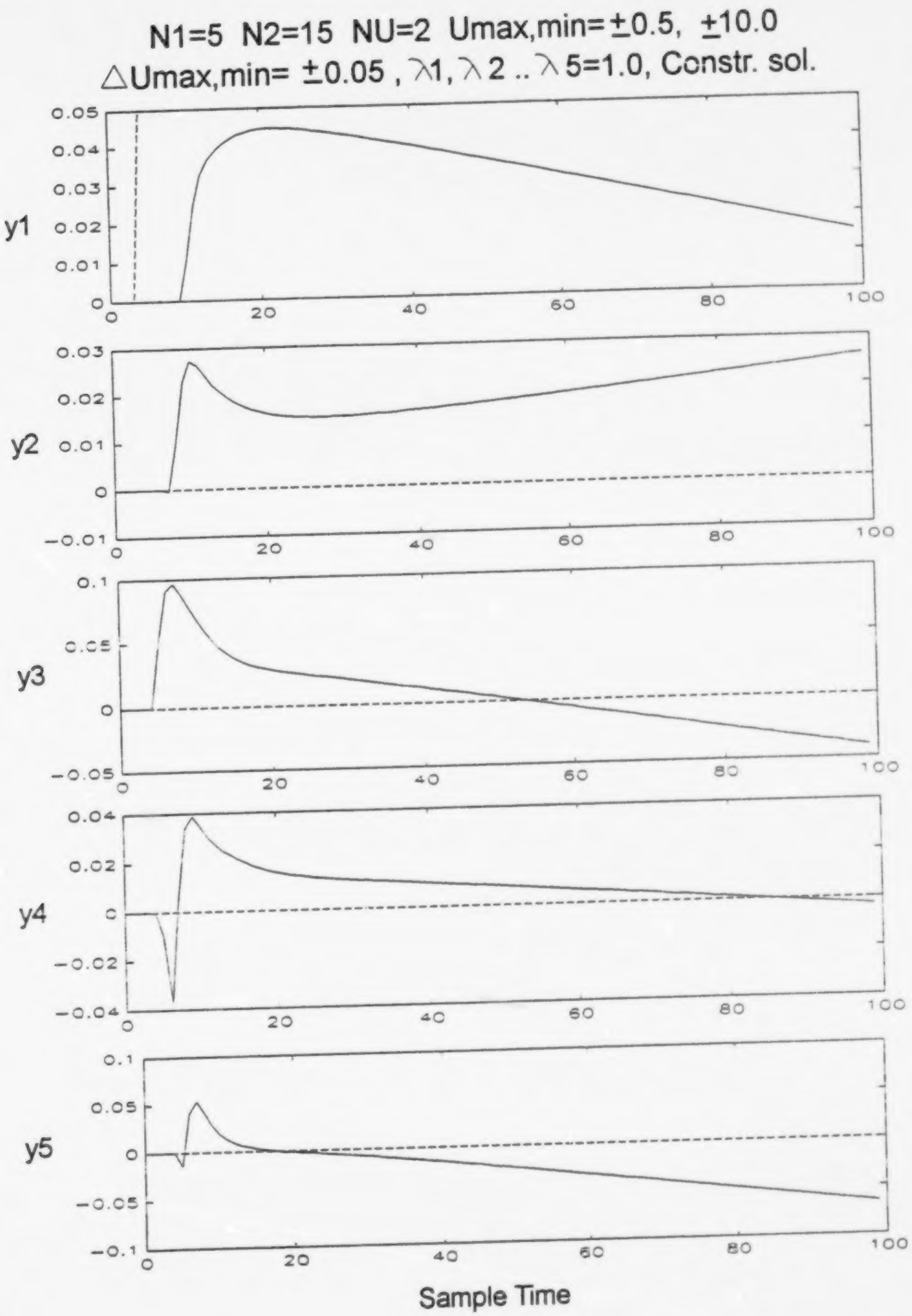


Figure 3.7 Simulation of a poorly conditioned system with λ weighting set to 1.0

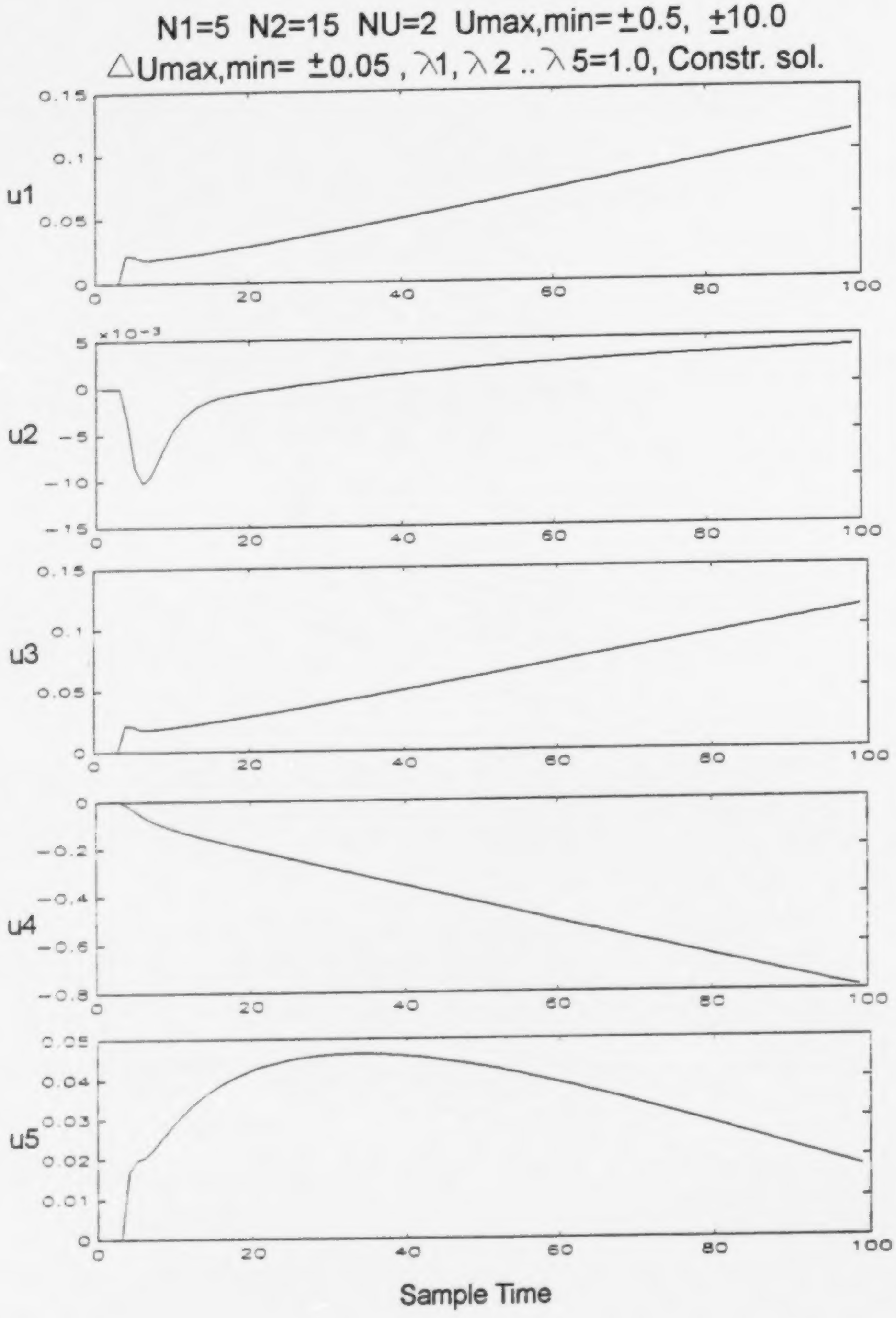


Figure 3.8 Simulation of a poorly conditioned system with λ weighting set to 1.0

output y1 only, thus this output needs maximum 'attention', which translates into maximum output weighting, while all other outputs can have equal but smaller weighting. Observations from this simulation (figures 3.9 and 3.10) are as follows:

- The drift in the outputs seen in figures 3.7 and 3.8 is eliminated, and the outputs converge to their respective set points slowly. The slow convergence is expected because of the large λ weighting of 20.0, which restricts the inputs from moving fast.

Further details of the effect of λ on the control action are available in section 2.2.2. Thus if the Hessian matrix of a system is poorly conditioned, it is desirable to increase the λ weighting and make the Hessian well conditioned.

3.3.3 The effect of numerical precision on simulation results

The numerical precision of a calculation depends on the number of digits which are retained when performing a calculation. Single precision calculations are done with 8 digits after the decimal point (in exponential notation). Thus if two numbers are to be subtracted which differ in the 9th decimal place, then the result of the subtraction will be 0, which is not correct. This sort of error can accumulate in situations where calculations are done in an iterative manner, *i.e.* where previous results are used to calculate future values. A good example of the way numerical precision effects simulations results is seen in figures 3.11 and 3.12. The simulation presented in these figures use the model in equation 3.7. These simulations are open loop responses to a 0.5 step change in the value of u1. From these figures a few observations can be made as follows:

- When single precision arithmetic is used, the outputs y1 and y2 reach a steady state, but this steady state is not what is expected. The expected steady state values of y1, y2, y3, y4 and y5 can be calculated from equation 3.7 by multiplying the gains in column 1 with 0.5. This results in the following steady state values:

  $y1_{ss} = 2.025$

  $y2_{ss} = 2.695$

  $y3_{ss} = 1.83$

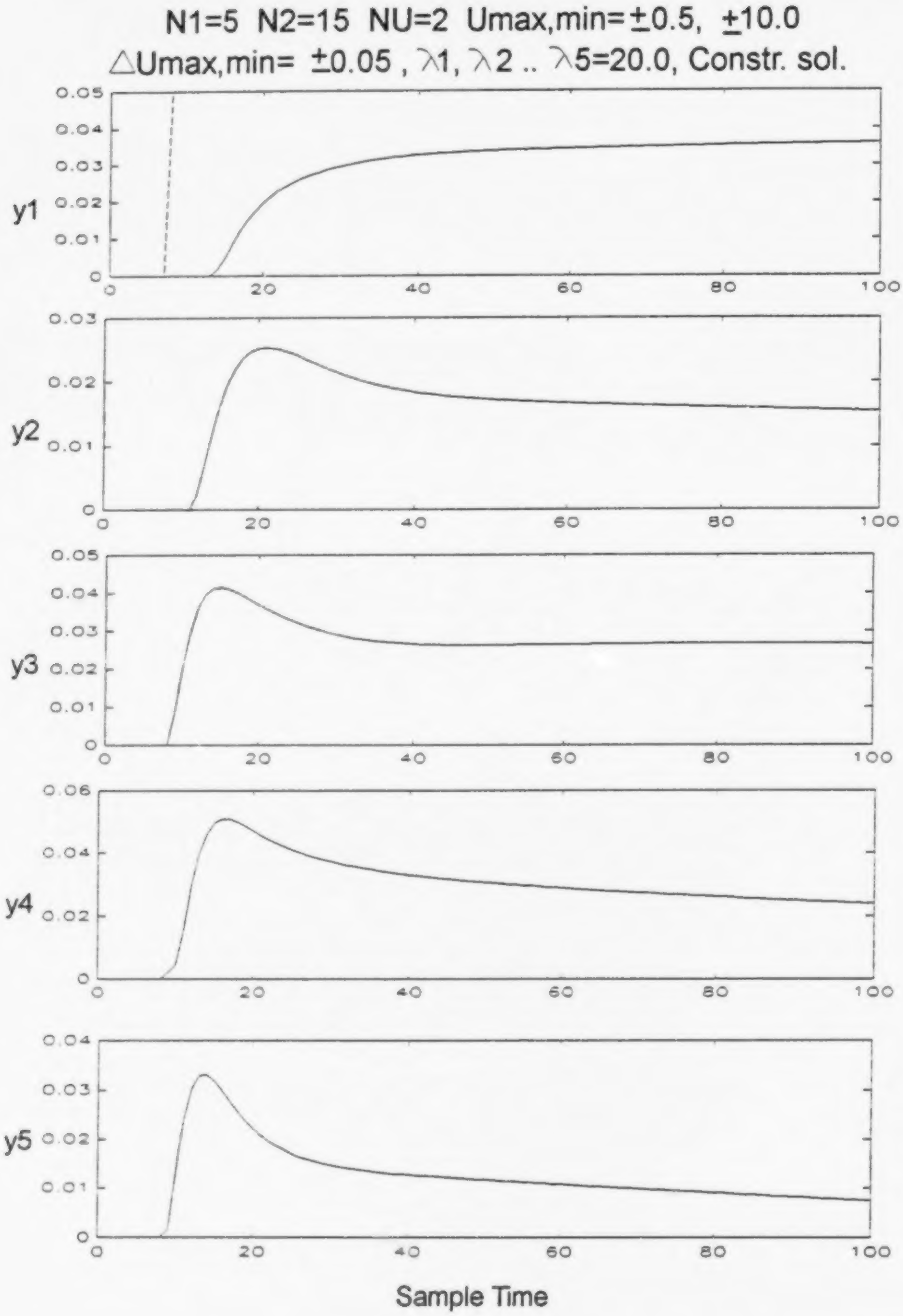


Figure 3.9 Simulation of a poorly conditioned system with λ weighting set to 20.0 and $\gamma_{1,2,3,4,5}$ = 1.0,0.5,0.5,0.5,0.5.

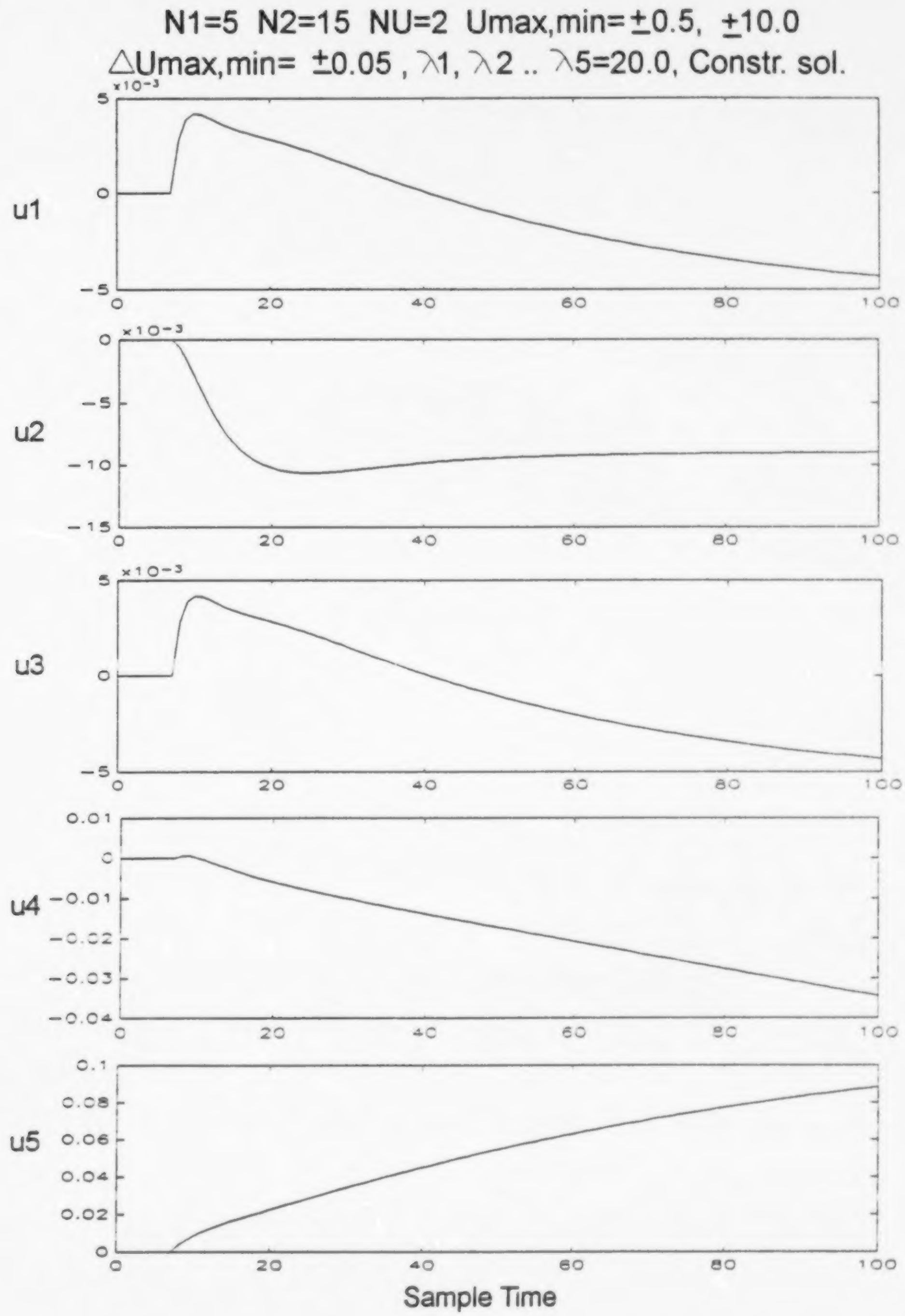


Figure 3.10 Simulation of a poorly conditioned system with λ weighting set to 20.0 and $\gamma_{1,2,3,4,5}$ = 1.0,0.5,0.5,0.5,0.5.

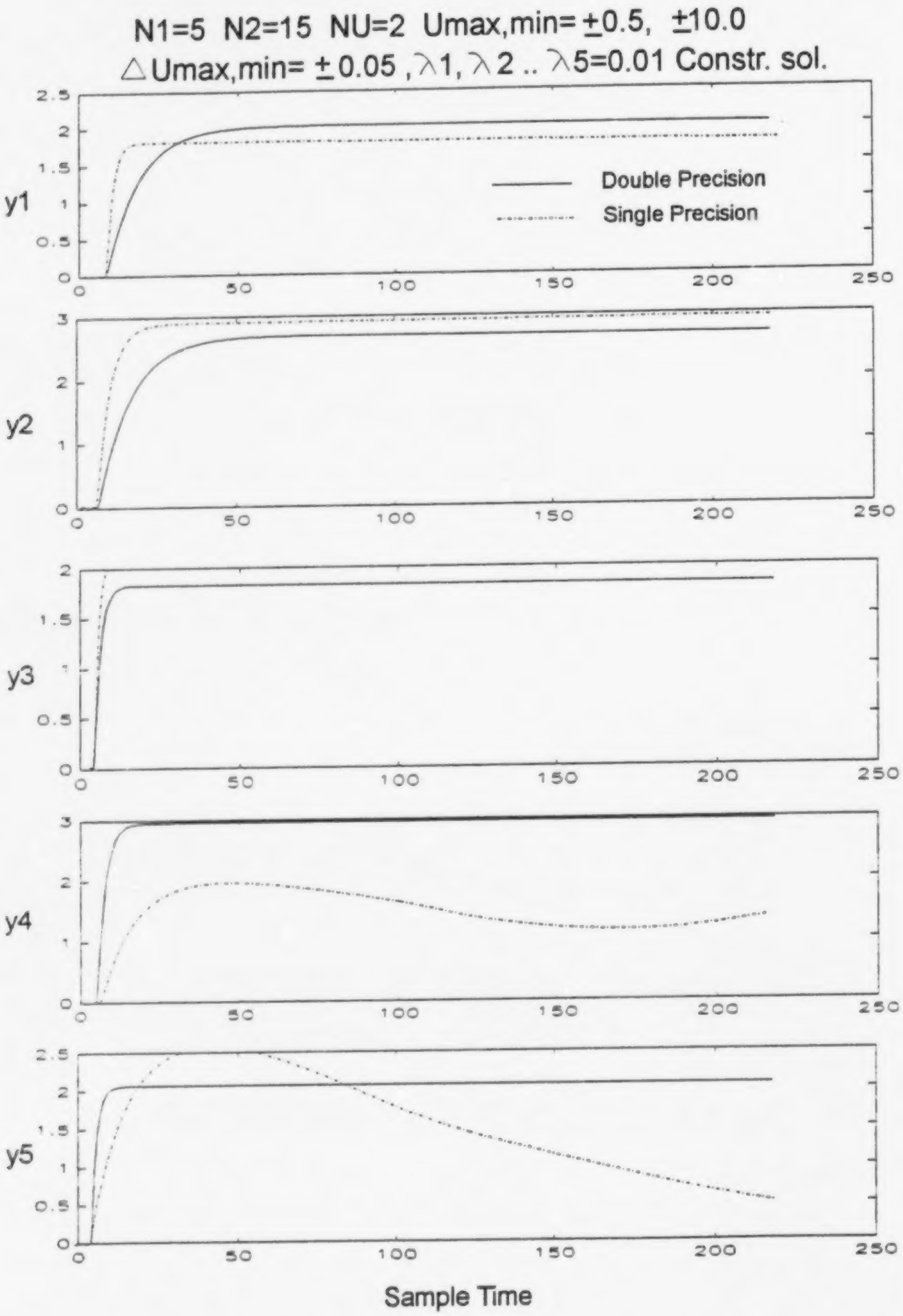


Figure 3.11 The effect of precision of variables on open loop simulations

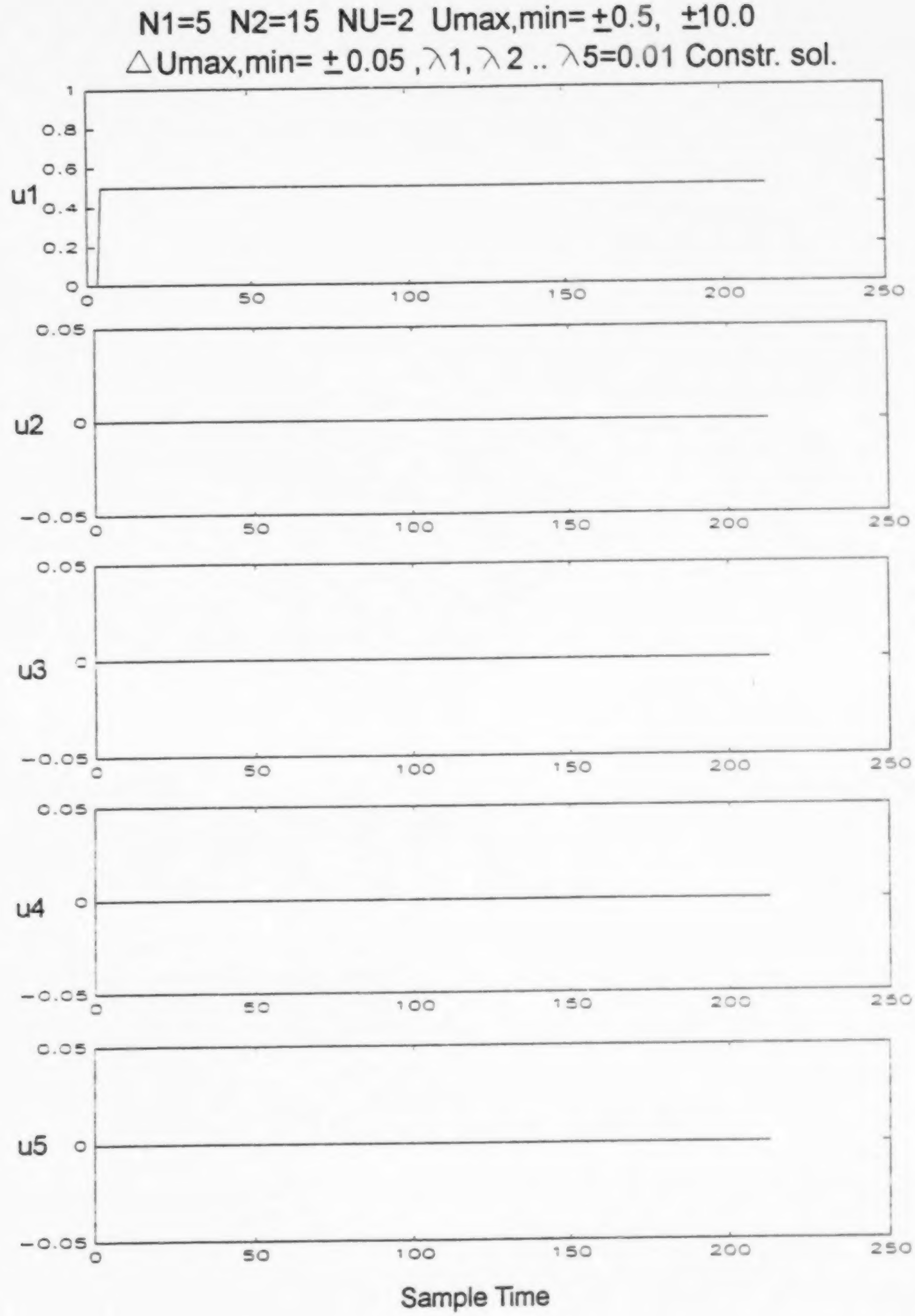


Figure 3.12 The effect of precision of variables on open loop simulations

$y4_{ss} = 2.96$

$y5_{ss} = 2.065$

- The outputs y4 and y5 for the single precision case deviate significantly from the expected steady state.
- When double precision arithmetic is used, all outputs converge to their respective steady states, consistent with the values calculated from equation 3.7.

In order to detect the cause of this error, a window of data was taken from the single precision simulation and further calculations were performed on this data by using MATLAB. It was found that the MATLAB simulation converged while the single precision simulation did not. Since MATLAB variables are double precision, it became obvious that the cause of error was mathematical precision. Furthermore, the variables which cause this error were identified as those which are used to calculate the output from the inputs.

For the complete GPC program to be compatible, all the 'C' programs were changed from single precision to double precision. The GSI (G2 Standard Interface) program was also changed to double precision. Double precision QPSOL FORTRAN programs were used instead of the single precision programs used before. For the simulation of a 2-input 2-output process, Mutha (1990) and Walther (1992) did not find similar numerical instability, however, the large dimension of the problem (equation 3.7) simulated in this work results in a significant increase in the number of arithmetic operations, making this simulation prone to such numerical problems.

The knowledge base developed on G2 did not have to be changed due to the change from single to double precision variables.This is because the precision of the variables used in the knowledge base is determined by the precision of the GSI variables, thus changing certain variables in the GSI program to double precision will automatically change their precision in the knowledge base.

3.4 Conclusions

In this chapter, an attempt is made to gain some insight into the structure and dynamics of large multivariable systems. The structure of the transfer function matrix provides insight on the complexity of interactions present in the process, and can be considered as the first step in analyzing the process dynamics. Subsequent manipulations to the transfer function matrix can be carried out so as to reach the 'simplest and most diagonally dominant' form of the transfer function matrix. Before proceeding with a simulation or an experimental evaluation of a control algorithm, it is extremely important that a step by step analysis is done to see the effect of certain parameters such as the sampling time, on the closed loop response of the process.

Choosing the correct sampling time is important in both single-input single-output or SISO processes as well as multi-input multi-output or MIMO processes. However, in the case of MIMO processes there can be multiple time scales, which causes ambiguity in selecting an appropriate sampling time. A simple *rule* is followed for selecting the appropriate sampling time for the process based on the dominant or highest time constant in the transfer function matrix. It should be noted that this rule does not take into consideration time delays which are extremely large (almost equal to the time constant of the process).

Another important issue addressed in this chapter is the use of a weighting factor λ for making a poorly conditioned Hessian matrix well conditioned. A poorly conditioned Hessian matrix will result in an erroneous Hessian inverse, which will be reflected in the control calculation as incorrect control action being calculated.

Simulations with single and double precision variables have shown that for the same set of conditions, the open loop simulation of a large MIMO system with single precision variables results in incorrect steady states and diverging outputs, while the simulation with double precision variables results in correct and stable steady states. It should be emphasised that these simulations were open loop, and all the transfer functions in the transfer function matrix were open loop stable. This problem of numerical stability can be considered example specific, however, the probability of such numerical instability occurring in large MIMO systems is far greater than that for smaller MIMO or SISO systems.

Chapter 4

Simulation of a Multivariable System

Most industrial processes, are multivariable in nature. Understanding the dynamics of these processes is key in selecting the control strategy to be implemented on these processes, and tuning it.

In this chapter, a benchmark problem, the Shell Control Problem, (Prett and Morari, 1986) is used to test the performance of the GPC tuning strategy.

In section 4.1 the problem is defined. In section 4.2 the control strategy and selection of initial tuning parameters is discussed and some closed loop simulations are carried out. Section 4.3 is devoted to the autotuning method and some simulations are carried out to illustrate the autotuning scheme. Conclusions are presented in section 4.4.

4.1 The Shell Control Problem[1]

The performance of Multivariable GPC is evaluated in this section by applying the algorithm to a widely used benchmark problem.

4.1.1 Process description

---

[1] The simulations were performed by discretizing the transfer function matrix in cascade with a zero order hold device using MATLAB functions.

This benchmark problem was put forward by Prett and Morari (1986). The process consists of a heavy oil fractionator column with three product draws and three side circulating loops. A schematic of the column is shown in figure 4.1 and a block diagram representation of the process is in figure 4.2. The feed to the column is gaseous and provides the necessary heat requirement of the column. Product specifications for the top and side draws are determined by economics and process requirements.

4.1.2 Process model

The process model consists of a matrix of transfer functions shown in table 4.1. All transfer functions in the Shell Control Problem are represented as first order with time delay as follows:

$$G = \frac{K\,e^{-\theta s}}{\tau s+1} \tag{4.1}$$

Where K is the gain of the transfer function, τ is the time constant, in minutes, and θ is the time delay in minutes as well. The corresponding uncertainties associated with these transfer functions are shown in table 4.2. Clearly, from table 4.1 it is evident that this process is poorly conditioned and has multiple time scales. The process is not diagonally dominant as well, making it even harder to control. The reader is referred to section 3.3 which discusses the issues of poorly conditioned, multiple time scales and diagonally dominant systems. It should be noted that for simulation purposes, the units of the time constants and delays are taken as seconds, rather than minutes. This is done to conserve the amount of time spent on simulations, since all simulations are done in real-time. The reason for real-time simulations is that the knowledge base is developed on G2, which has a real-time clock with the smallest resolution of 1 second.

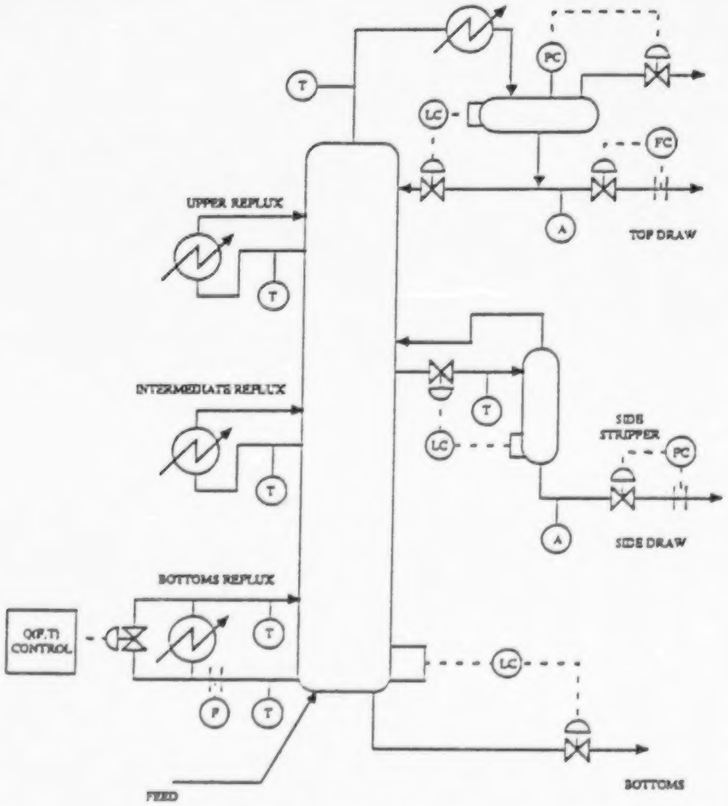


Figure 4.1 Schematic of Shell heavy oil fractionator

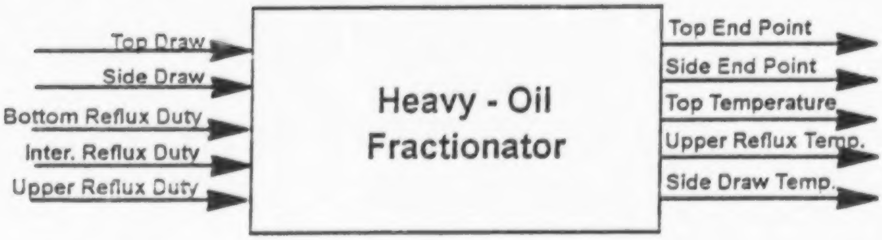


Figure 4.2 Block diagram of heavy oil fractionator

	Top Draw	Side Draw	Bottoms Reflux Duty	Inter. Reflux Duty	Upper Reflux Duty
Top End Point	$K = 4.05$ $\tau = 50$ $\theta = 27$	$K = 1.77$ $\tau = 60$ $\theta = 28$	$K = 5.88$ $\tau = 50$ $\theta = 27$	$K = 1.20$ $\tau = 45$ $\theta = 27$	$K = 1.44$ $\tau = 40$ $\theta = 27$
Side End Point	$K = 5.39$ $\tau = 50$ $\theta = 18$	$K = 5.72$ $\tau = 60$ $\theta = 14$	$K = 6.90$ $\tau = 40$ $\theta = 15$	$K = 1.52$ $\tau = 25$ $\theta = 15$	$K = 1.83$ $\tau = 20$ $\theta = 15$
Top Temp.	$K = 3.66$ $\tau = 9$ $\theta = 2$	$K = 1.65$ $\tau = 30$ $\theta = 20$	$K = 5.53$ $\tau = 40$ $\theta = 2$	$K = 1.16$ $\tau = 11$ $\theta = 0$	$K = 1.27$ $\tau = 6$ $\theta = 0$
Upper Reflux Temp.	$K = 5.92$ $\tau = 12$ $\theta = 11$	$K = 2.54$ $\tau = 27$ $\theta = 12$	$K = 8.10$ $\tau = 20$ $\theta = 2$	$K = 1.73$ $\tau = 5$ $\theta = 0$	$K = 1.79$ $\tau = 19$ $\theta = 0$
Side Draw Temp.	$K = 4.13$ $\tau = 8$ $\theta = 5$	$K = 2.38$ $\tau = 19$ $\theta = 7$	$K = 6.23$ $\tau = 10$ $\theta = 2$	$K = 1.31$ $\tau = 2$ $\theta = 0$	$K = 1.26$ $\tau = 22$ $\theta = 0$
Inter. Reflux Temp.	$K = 4.06$ $\tau = 13$ $\theta = 8$	$K = 4.18$ $\tau = 33$ $\theta = 4$	$K = 6.53$ $\tau = 9$ $\theta = 1$	$K = 1.19$ $\tau = 19$ $\theta = 0$	$K = 1.17$ $\tau = 24$ $\theta = 0$
Bottoms Reflux Temp.	$K = 4.38$ $\tau = 33$ $\theta = 20$	$K = 4.42$ $\tau = 44$ $\theta = 22$	$K = 7.20$ $\tau = 19$ $\theta = 0$	$K = 1.14$ $\tau = 27$ $\theta = 0$	$K = 1.26$ $\tau = 32$ $\theta = 0$

Table 4.1 Transfer functions of the Shell Problem

As part of the problem specification, uncertainties in the gains of the process transfer functions are also considered. These uncertainties are used to evaluate the performance of the system under model plant mismatch conditions, in a later section. Since in reality, industrial plants can never be represented accurately by a model, the issue of a mismatch between models and plants is important and realistic. The closed loop performance and the tuning strategy of GPC are also affected by model plant mismatch. The matrix of uncertainties in the gains of transfer functions of the Shell Control Problem is represented in the following table where:

$$-1 \leq \varepsilon_i \leq 1 \qquad i = 1,2,3,4,5$$

	Top Draw	Side Draw	Bottoms Reflux Duty	Inter. Reflux Duty	Upper Reflux Duty
Top End Point	$4.05 + 2.11\varepsilon_1$	$1.77 + 0.39\varepsilon_2$	$5.88 + 0.59\varepsilon_3$	$1.20 + 0.12\varepsilon_4$	$1.44 + 0.16\varepsilon_5$
Side End Point	$5.39 + 3.29\varepsilon_1$	$5.72 + 0.57\varepsilon_2$	$6.90 + 0.89\varepsilon_3$	$1.52 + 0.13\varepsilon_4$	$1.83 + 0.13\varepsilon_5$
Top Temp	$3.66 + 2.29\varepsilon_1$	$1.65 + 0.35\varepsilon_2$	$5.53 + 0.67\varepsilon_3$	$1.16 + 0.08\varepsilon_4$	$1.27 + 0.08\varepsilon_5$
Upper Reflux Temp.	$5.92 + 2.34\varepsilon_1$	$2.54 + 0.24\varepsilon_2$	$8.10 + 0.32\varepsilon_3$	$1.73 + 0.02\varepsilon_4$	$1.79 + 0.04\varepsilon_5$
Side Draw Temp	$4.13 + 1.71\varepsilon_1$	$2.38 + 0.93\varepsilon_2$	$6.23 + 0.30\varepsilon_3$	$1.31 + 0.03\varepsilon_4$	$1.26 + 0.02\varepsilon_5$
Inter. Reflux Temp.	$4.06 + 2.39\varepsilon_1$	$4.18 + 0.35\varepsilon_2$	$6.53 + 0.72\varepsilon_3$	$1.19 + 0.08\varepsilon_4$	$1.17 + 0.01\varepsilon_5$
Bottoms Reflux Temp.	$4.38 + 3.11\varepsilon_1$	$4.42 + 0.73\varepsilon_2$	$7.20 + 1.33\varepsilon_3$	$1.14 + 0.18\varepsilon_4$	$1.26 + 0.18\varepsilon_5$

Table 4.2 Uncertainties in the gains, K, given in table 4.1

4.1.3 Control objectives and constraints

The control objectives can be listed as follows:

1. Maintain the top and side draw product end points at specification (0.0 ± 0.005 at steady state).

2. Maximize steam make in the steam generators.

3. Reject unmeasured disturbances entering the column from the upper and intermediate refluxes due to change in heat duty requirements from other columns.

The following control constraints are also specified:

1. All draws must be within a hard maximum and minimum bounds of 0.5 and -0.5.

2. The bottom reflux duty is constrained within hard maximum and minimum bounds of ±0.5.

3. All manipulated variables have a maximum rate constraint of 0.05 per minute (0.05 per second for this work).

4. Fastest sampling time is 1 minute (1 second for this work).

5. Bottom reflux draw temperature has a minimum value of -0.5.

6. The top end point must be maintained within the maximum and minimum constraints of 0.5 and -0.5.

4.1.4 Open loop simulation

In order to understand the dynamics of this process, several open loop simulations are necessary. For the purposes of this work, a 5-input 5-output subset of the original 5-input, 7-output problem described in table 4.1 was considered. The original 5-input 7-output system is an example of an *underspecified* system (see section 3.2), which is not controllable. In order to reveal the interactions present in this complex process, the open loop simulations were performed by introducing a step change of magnitude 0.5 in inputs 1 to 5, one at a time. The intermediate reflux temperature and bottoms reflux temperature (last 2 rows from table 4.1) were not included in the simulations. The open loop simulation results are presented in figure 4.3. From this figure it is evident that the process interactions are very intense and decoupling these interactions can be a tedious task.

The sampling time for this system was chosen based on the rule of thumb: *choose a sampling time for the system such that it is approximately 1/10th the dominant time constant.* Thus a sampling time of 5 seconds was chosen.

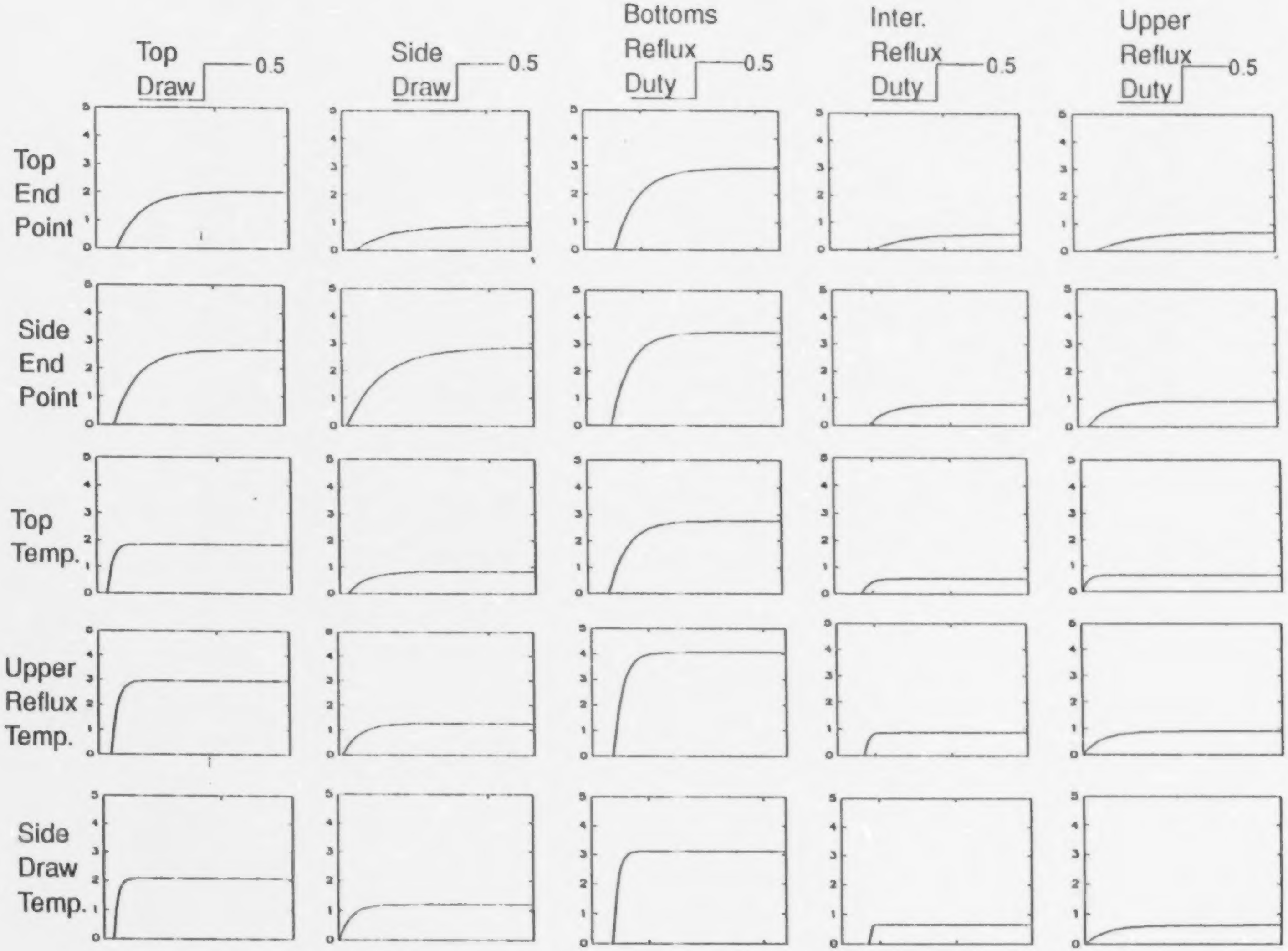


Figure 4.3 Step response matrix of Shell Problem

4.2 Control Strategy

The Generalised Predictive Control algorithm discussed in section 2.2 is used to control this process. This algorithm was chosen due to the nature of the fairly "large" 5-input 5-output problem which involved constraints, and consisted of intense interactions among different outputs. Since GPC is a multivariable control strategy, the control trajectory calculated will guarantee minimum interactions among different outputs, although these interactions cannot be eliminated altogether.

4.2.1 Selection of initial tuning parameters

In order to tune GPC, initial estimates of parameters are required. The choice of these initial parameters is the subject of discussion of this section.

To refresh the readers mind, tuning parameters of GPC are:

- N1, the minimum prediction horizon.
- N2, the maximum prediction norizon.
- NU, the control horizon.
- λ, the control weighting parameter.
- γ, the output weighting parameter.

The *initial* choice of these parameters can be made keeping the following *Rules* in mind.

Rule 1 *Choose N1 equal to the maximum delay in the process.*

Rule 2 *Choose N2 equal to 10+N1.*

Rule 3 *Choose NU equal to 1 if interactions in the process are not significant, or choose NU > 1 for processes with high interactions.*

Rule 4 *Set initial values of the control weighting parameter,* λ_i *, to a small number, say 0.01.*

Rule 5 *Set initial values of the output weighting parameter,* γ_i *, to 1.0.*

Using the above *rules* results in the selection of the following initial tuning parameters:

N1 = 5

N2 = 15

NU = 2

$\lambda_i = 0.01$ i=1,2,3,4,5

$\gamma_i = 1.0$ i=1,2,3,4,5

It should be noted that these rules are based on experience of the author with large (5-input, 5-output) Multivariable systems. Some single-input single-output analysis (SISO) done by McIntosh, *et al.* (1991) and experience of Mutha (1990), Walther (1992) in tuning Multivariable GPC is also used in choosing the above mentioned initial tuning parameters.

4.2.2 Closed loop simulations

In order to validate the choice of N1, N2 and NU, a few simulations were performed to see the affect of N1 and NU on the closed loop response of the system. It should be noted that no autotuning is carried out in these cases. All the parameters are set manually at the beginning of the simulation. For purposes of comparison, the base case has N1=5, N2=15, NU=2, $\lambda_{1,2,3,4,5} = 0.01$, $\gamma_{1,2,3,4,5} = 1.0$ and the solution is unconstrained. An observation made by Walther (1992) is that for NU > 1 , choosing different N2 's does not affect the closed loop response of the process, as long as the criteria $N2\text{-}NU \geq d_{max}\text{-}1$, $N1 < d_{max}$ established by Shah *et al.* (1987) is satisfied. In case the maximum delay, $d_{max} \leq N1$, then d_{max} in the above inequality can be replaced by N1(*i.e.* $N2\text{-}NU \geq N1\text{-}1$). Since setting N2 to 15 will satisfy the criteria, for N1=1,2,3,4 or 5 and NU=1 or 2, thus the value of N2 is fixed at 15. The following simulations were carried out :

#1 N1=1, N2=15, NU=1, $\lambda_{1,2,3,4,5} = 0.01$, $\gamma_{1,2,3,4,5} = 1.0$

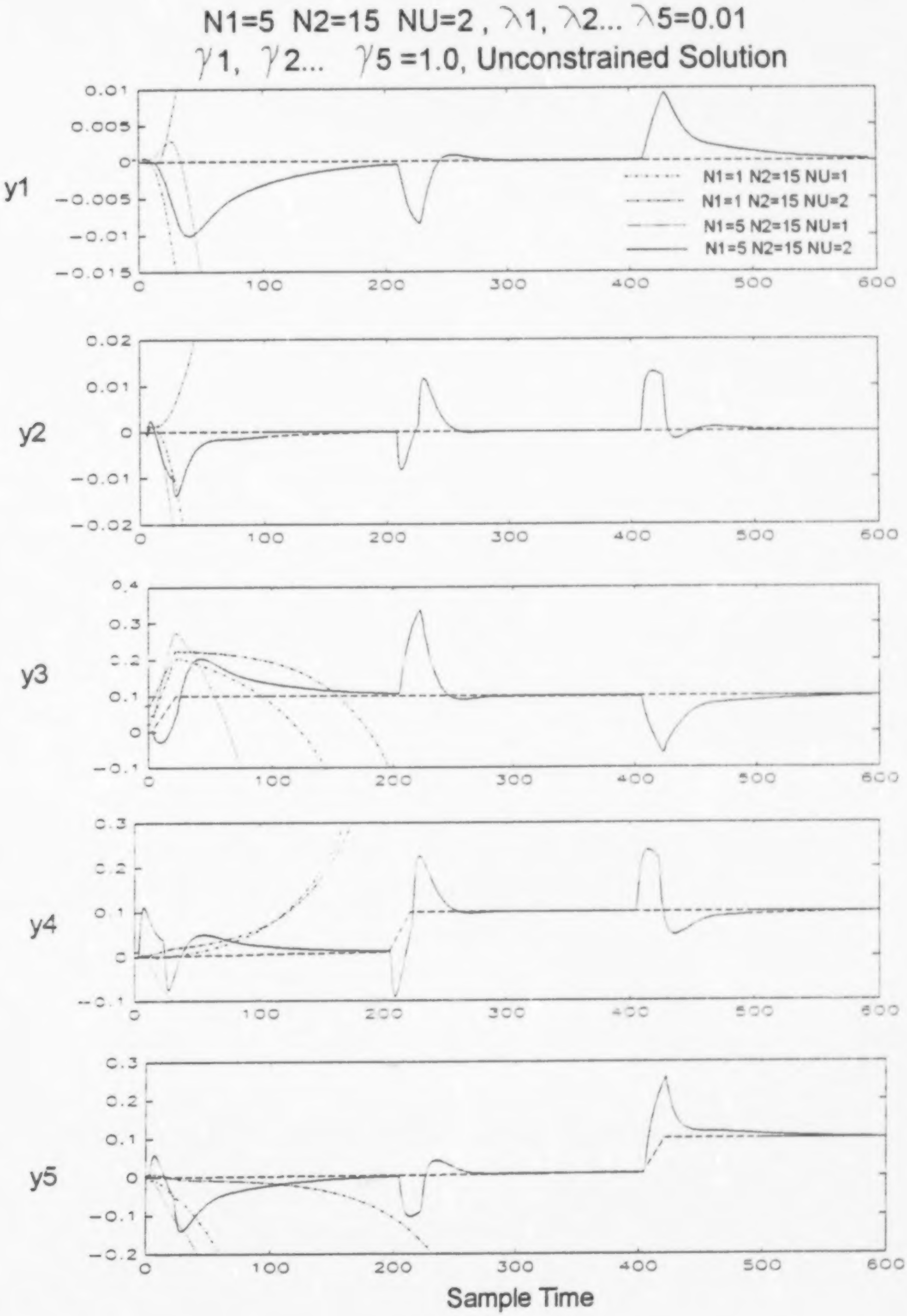
#2 N1=1, N2=15, NU=2, $\lambda_{1,2,3,4,5} = 0.01$, $\gamma_{1,2,3,4,5} = 1.0$

#3 N1=5, N2=15, NU=1, $\lambda_{1,2,3,4,5} = 0.01$, $\gamma_{1,2,3,4,5} = 1.0$

#4 N1=5, N2=15, NU=2, $\lambda_{1,2,3,4,5} = 0.01$, $\gamma_{1,2,3,4,5} = 1.0$

From figures 4.4 and 4.5, we see that the only stable and offset free closed loop response is given by simulation #4, validating the *rules* established and the initial tuning parameters chosen in section 4.2.1. Some additional observations are as follows:

- From figure 4.5 we observe that u1 and u4 for simulation #'s 1, 2 and 3 immediately go off scale. This clearly indicates the sensitivity of the closed loop



N1=5 N2=15 NU=2 , λ1, λ2... λ5=0.01
γ1, γ2... γ5 =1.0, Unconstrained Solution

Figure 4.4 Choosing initial tuning parameters for GPC

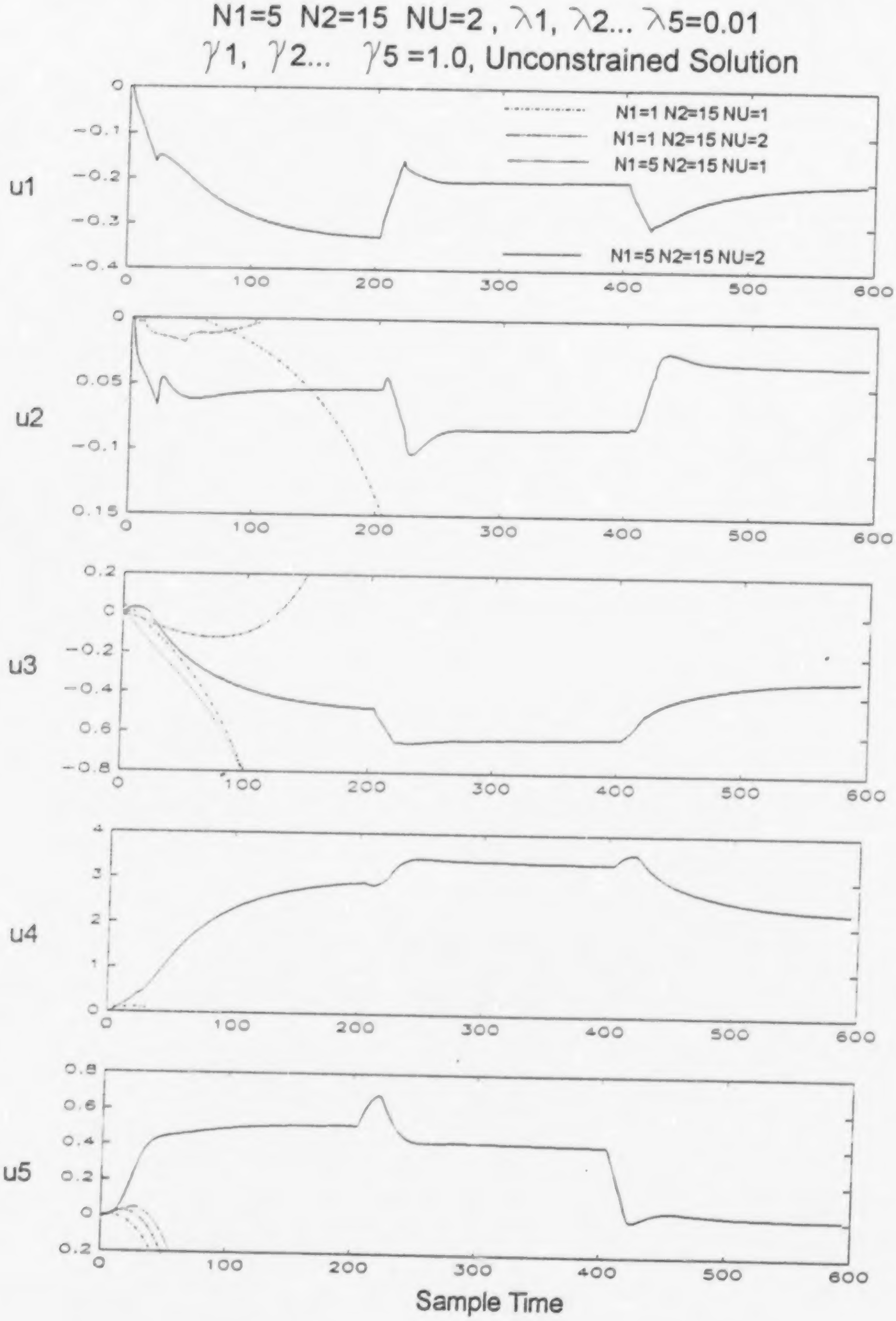


Figure 4.5 Choosing initial tuning parameters for GPC

system to the choice of N1 and NU.

- The 'default GPC tuning parameters' of N1=1+delay, N2=10 to 20 and NU=1 for single-input, single-output and 'smaller' multivariable systems is not applicable in this case due to large time delays and intense interactions.

Increasing the output weighting, λ, results in a more gradual control trajectory, consequently a slower response. Results which describe the effect of varying λ for a MIMO system are well established by Mutha (1990). Details on the effect of choosing λ appropriately in combination with a T-filter, in the case of model plant mismatch, are available in section 4.3.4. For a SISO system, McIntosh *et al.* (1991) showed that the closed loop poles converge to the open loop poles when N2 is made extremely large, which translates into a slowing of the response as N2 increase. The same behaviour was observed by Walther (1992) for 2-input 2-output MIMO process, with NU set to 1, confirming, in part, the observations of McIntosh and co-workers.

4.3 Autotuning GPC on the Shell Problem

The objective of this section is to test the applicability of the tuning rules developed in section 2.4.4 on the Shell Control Problem. The tuning strategies evaluated are:

#1 Output or γ weighting tuning.

#2 T-filter tuning.

Another strategy used to enhance control performance is the constrained output strategy. Although this is not a tuning method, it is an important strategy in constraining the outputs within given hard bounds.

In order to determine the impact of each tuning method, a base case scenario is established here as well. The base case has the following tuning parameters:

N1 = 5

N2 = 15

NU = 2

$\lambda_i = 0.01$ i=1,2,3,4,5

$\gamma_i = 1.0$ i=1,2,3,4,5

T-filter = $1-0.8q^{-1}$

In addition to the above parameters, other parameters due to constraints as discussed in section 4.1.3 are as follows:

$\Delta u_{1,2,3,4,5} = \pm\ 0.05$ per sampling interval.

$u_{1,2,3} = \pm 0.5$

$u_{4,5} = \pm 10.0$

The regulation band for the outputs y1, y2, y3, y4 and y5 is ±0.1, *i.e.* the outputs should be kept within ±0.1 of their set point. The rate constraint and amplitude constraint parameters are used as part of the problem formulation and at each sampling instant, a set of inputs u1, u2, u3, u4 and u5 are calculated by QPSOL, a constrained quadratic programming algorithm. Details of QPSOL are in section 2.2.5.3.

A simulation of the process with the above parameters is presented in figures 4.6 and 4.7. A few observations are as follows:

- When the first set point (dashed line) change is made in y3, all the outputs are affected. After the initial transient they all move towards the set point until sampling time t=150. At this time, inputs u1 and u3 (figure 4.7) hit the constraint of -0.5, thus resulting in a sudden change in outputs and an offset at steady state.
- A set points in output y4 at times t=200 is introduced as well. Here too, after the initial transient, between sampling time 220 and 400, there is an offset in all the outputs (figure 4.6). This offset is due to the input u3 (figure 4.7) saturating at the lower constraint of -0.5.
- A final set point change is made at time t=400. Interactions appear in all other outputs immediately but in this case, all the outputs reach their set point and all the inputs are in the feasible region *i.e.* none of them are saturated at any of the constraints.

These observations bring to light the idea that there are certain *infeasible* set points which cannot be tracked given constraints on inputs.

4.3.1 Output weighting tuning

Initial GPC tuning parameters for the 5-input 5-output Shell Control Problem were

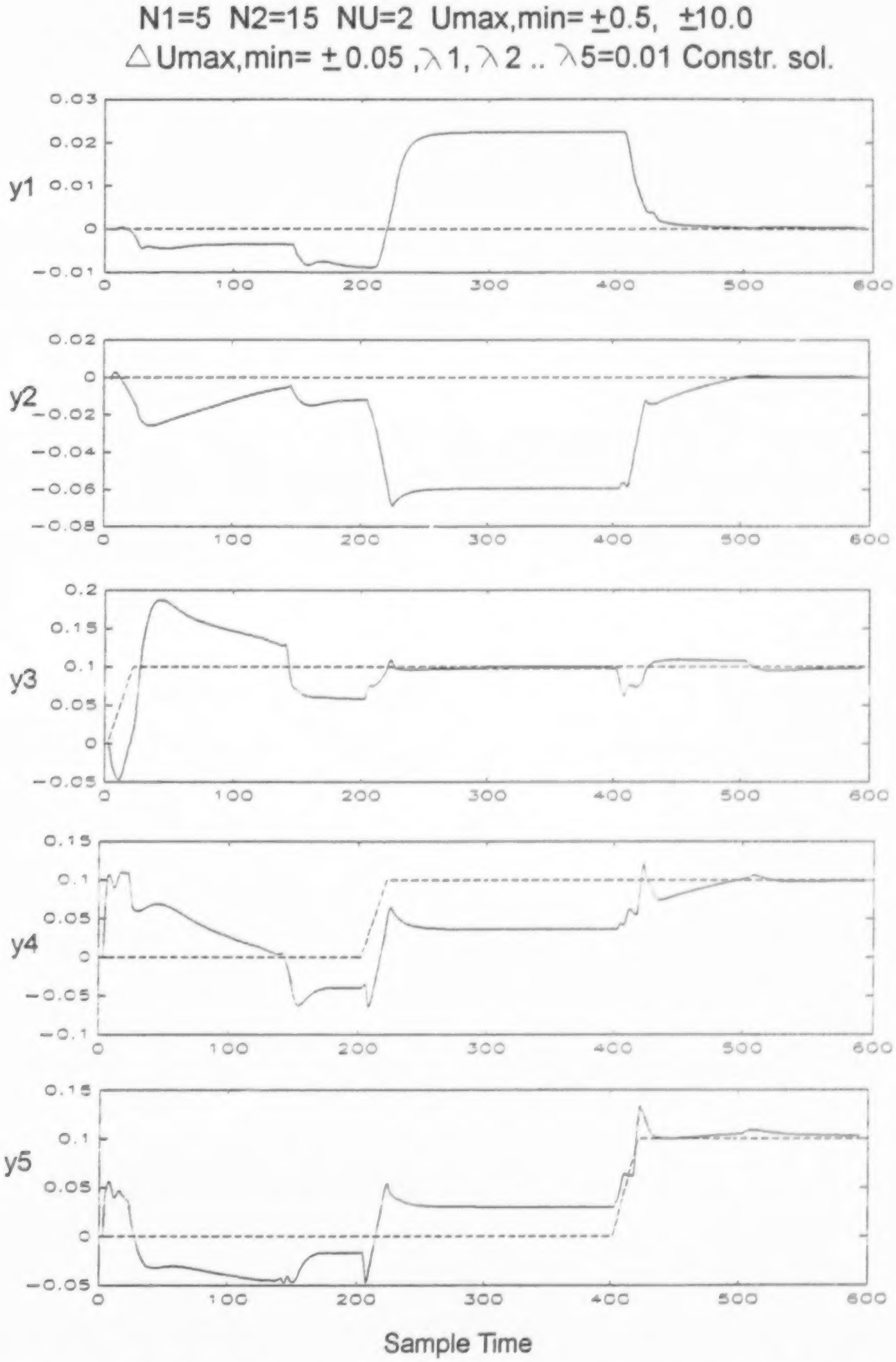


Figure 4.6 Closed loop simulation without autotuning and output constraints

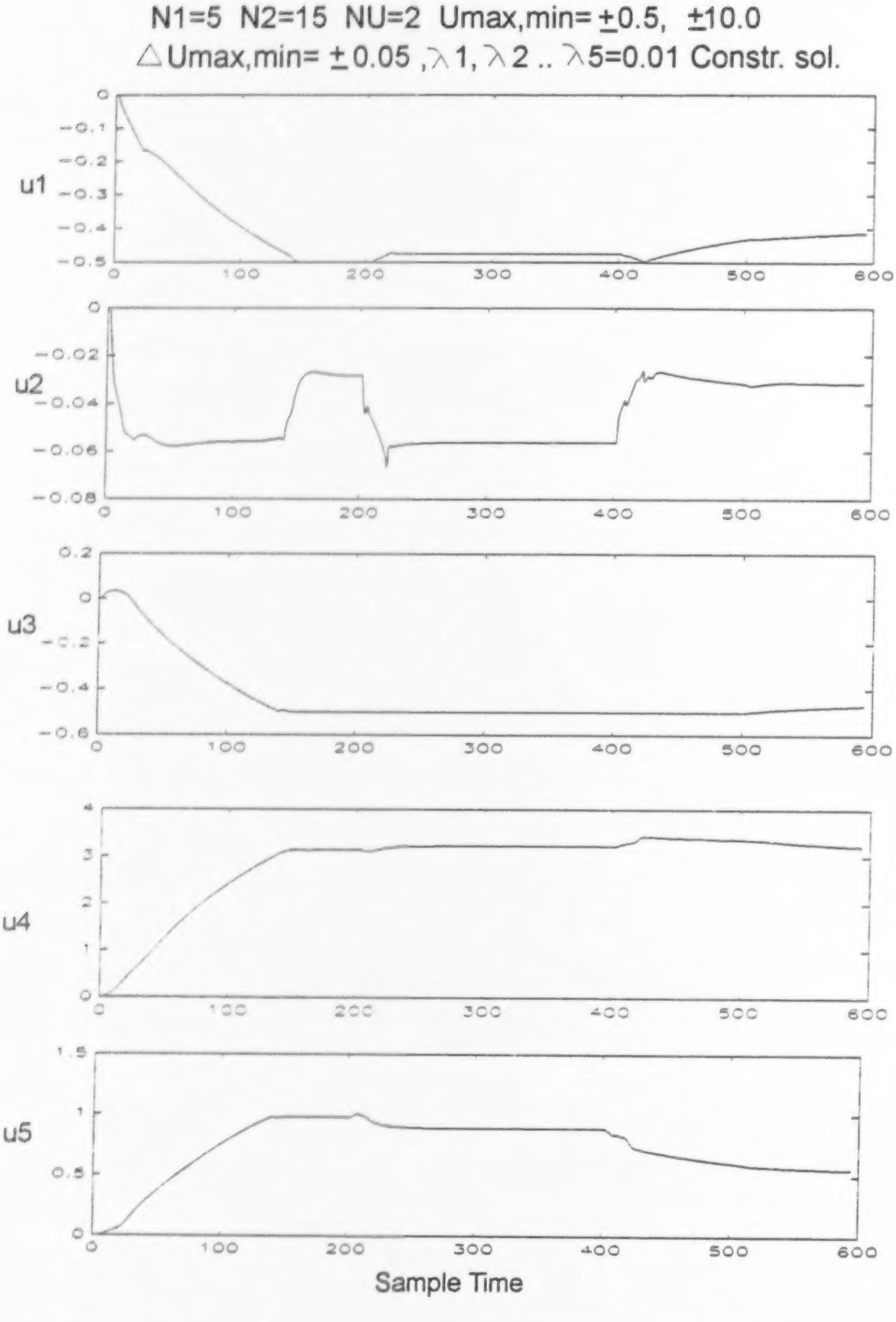

N1=5 N2=15 NU=2 Umax,min= ±0.5, ±10.0
△Umax,min= ± 0.05 ,λ1, λ2 .. λ5=0.01 Constr. sol.

Figure 4.7 Closed loop simulation without autotuning and output constraints

decided in section 4.2.1. However, due to the multivariable nature of the process, certain parameters such as output weighting, if tuned at regular intervals, can help decouple the outputs and improve the overall closed loop performance of the process. This section illustrates the applicability of output weighting tuning rules developed by Walther (1992), on the Shell problem. The reader is referred to section 2.4.4 for details of these tuning rules.

The simulation to illustrate the use of output weighting is in figures 4.8 and 4.9. This simulation is compared with the base case referred to above (figures 4.6 and 4.7). The following observations are made:

- The saturation of u1 and u3 at the constraint of -0.5 at sampling time 150 as observed in figure 4.7 does not take place in figure 4.9. As a result of this, all the outputs converge to the set points, rather than deviate from the set point due to saturated inputs. Thus tuning output weighting helps in improving closed loop performance.
- The saturation of u3 between sample time 205 and 410 is observed in figure 4.9 as well (compare with figure 4.7), meaning that the set point is *infeasible*, due to the constraints on the inputs. However, the output weighting does get set to values such that the offset is reduced. Comparing figures 4.8 and 4.6 shows that between sample time 205 and 410 the offset in outputs y1 and y2 for the tuned output weighting case is lower than that for the fixed output weighting case. For y4 and y5, the offset is the same for the tuned and fixed output weighting case. However, the offset for the tuned case is more than the fixed output weighting case in output y3.

The output weighting strategy thus produces *tradeoffs* between outputs and results in better overall closed loop control than when fixed output weighting is used.

4.3.2 T-filter tuning

The purpose of the T-filter is to make the closed loop robust to external disturbances and reduce noise in the inputs, and consequently the outputs. The tuning strategy for the T-filter is discussed in section 2.4.4.2. The effect of tuning the T-filter by itself is discussed

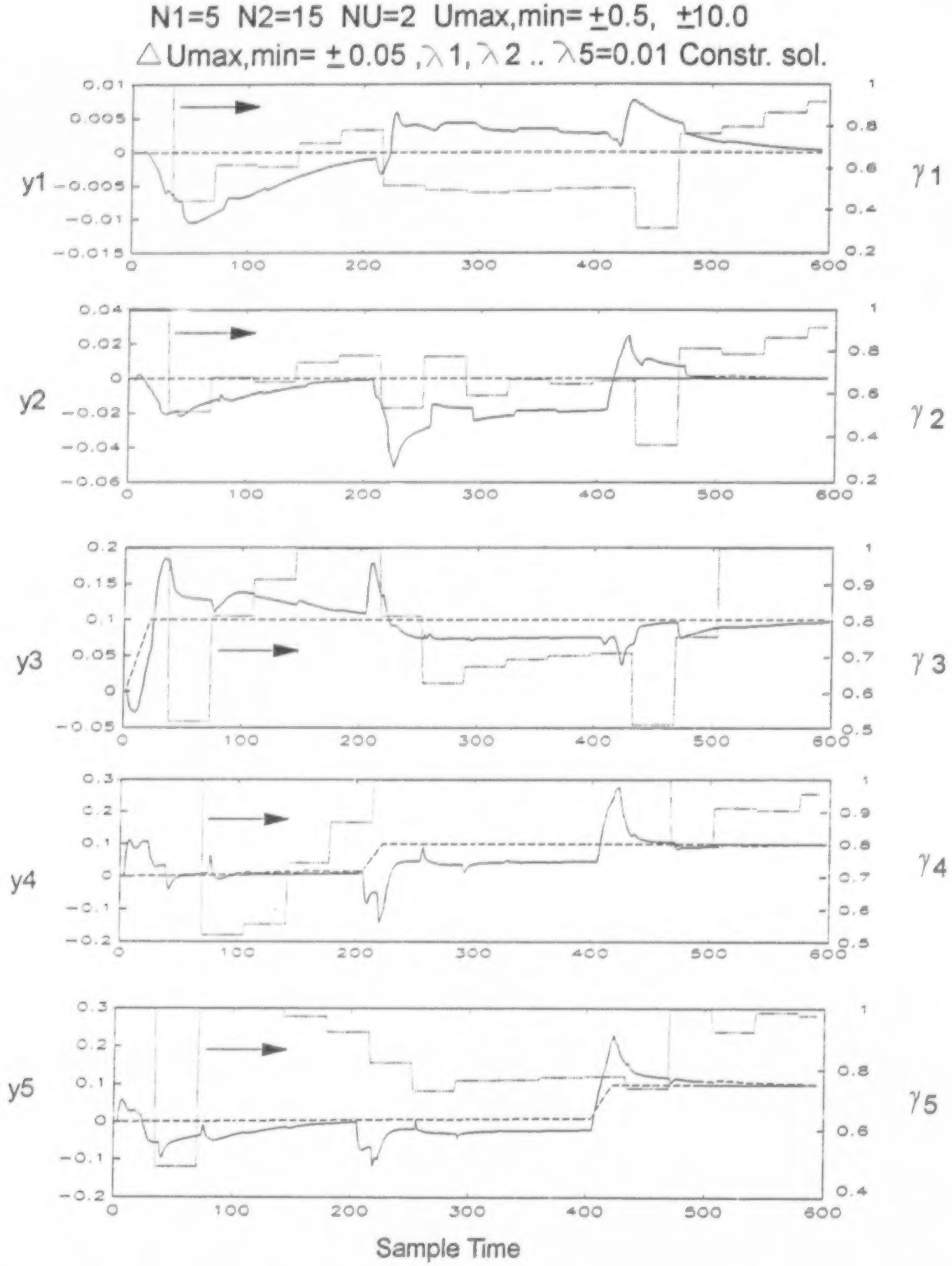


Figure 4.8 Closed loop simulation with tuned output weighting

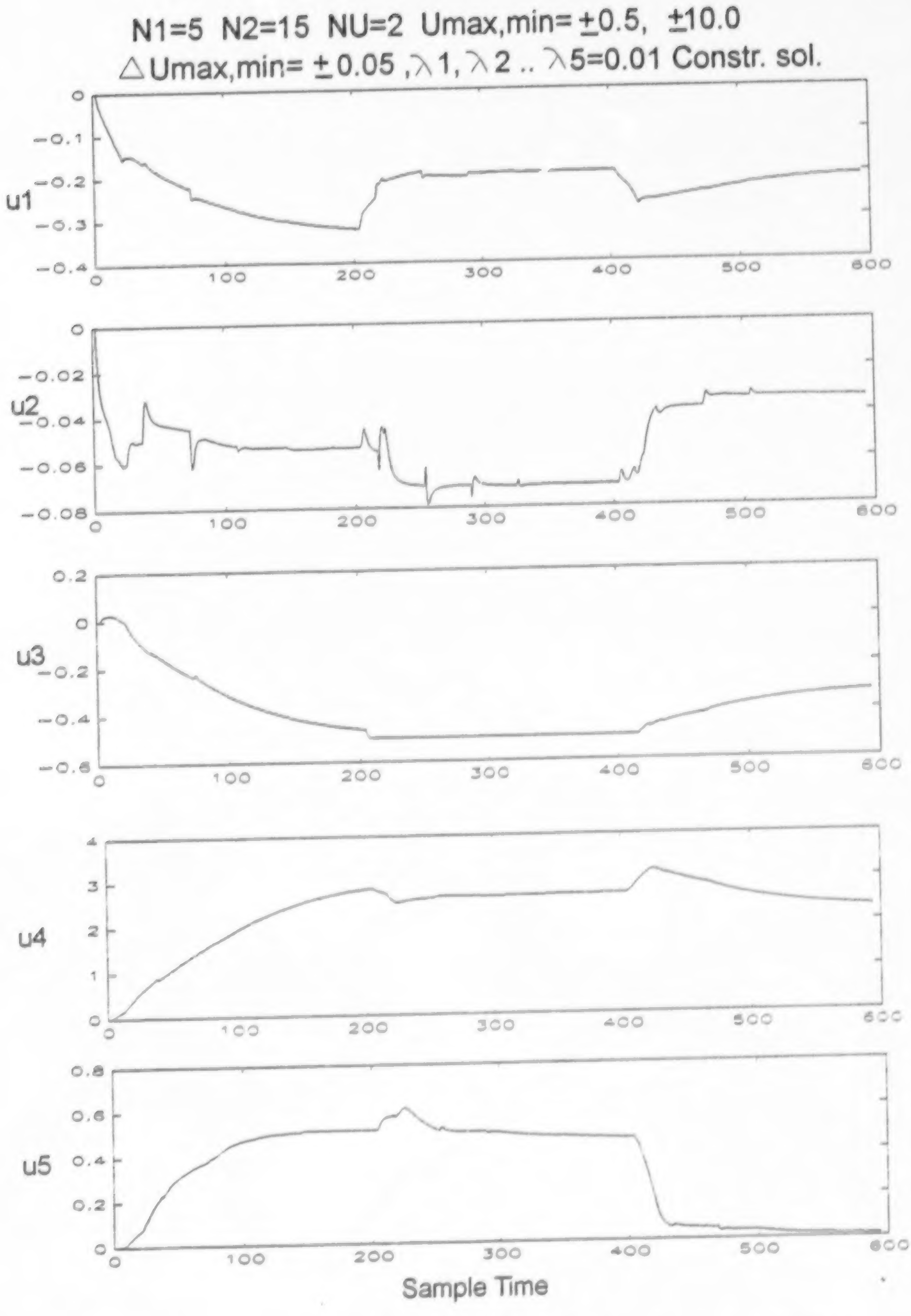


Figure 4.9 Closed loop simulation with tuned output weighting

in this section. The tuning parameters are the same as for the base case presented in figures 4.6 and 4.7, with the exception of the T-filter being tuned, rather than fixed at $1-0.8q^{-1}$.

From figures 4.10 and 4.11, we note the following:

- There is a 'spike' in the outputs at sample time 30. This is caused by the first 'jump' of the T-filter set automatically to the highest first order T-filter of $1-0.95q^{-}$ from the default value of $1-0.8q^{-1}$. This happens because the T-filter tuning rule detects a high variance in the inputs and consequently calculates this T-filter of $1-0.95q^{-1}$.
- The T-filter gets set to a second order filter at sample time 60. Comparing the response of the outputs in figure 4.10 after this time with the base case in figure 4.6 shows that the offset in y1 is reduced between time 210 and 400 in the tuned T-filter case as compared with the fixed T-filter case. This is due to the input u1 not saturating at the lower limit of -0.5 (figure 4.11) when the T-filter is tuned, as compared with u1 in figure 4.7 when u1 saturates at -0.5 at sample time 140.
- The final T-filter implemented is a second order filter, resulting in a better closed loop performance. From an off-line, decision support perspective, this technique can be used for deciding and fixing the type of T-filter required for a particular process, rather than use trial and error or ad-hoc methods to decide on an appropriate T-filter.

The use of this T-filter tuning strategy is more apparent when model plant mismatch is introduced in the process and the idea of increasing robustness and reducing noise is illustrated. These results are presented in a later section on model plant mismatch.

4.3.3 Output constraints

Another useful method to improve the closed loop performance of a process is to activate constraints on the output of the process. These constraints become a part of the problem formulation of GPC and the resulting quadratic programming problem is solved by QPSOL. Two simulations were performed in this case, one with output constraints only (figure 4.12 and 4.13) and the other with output constraints and tuned output weighting

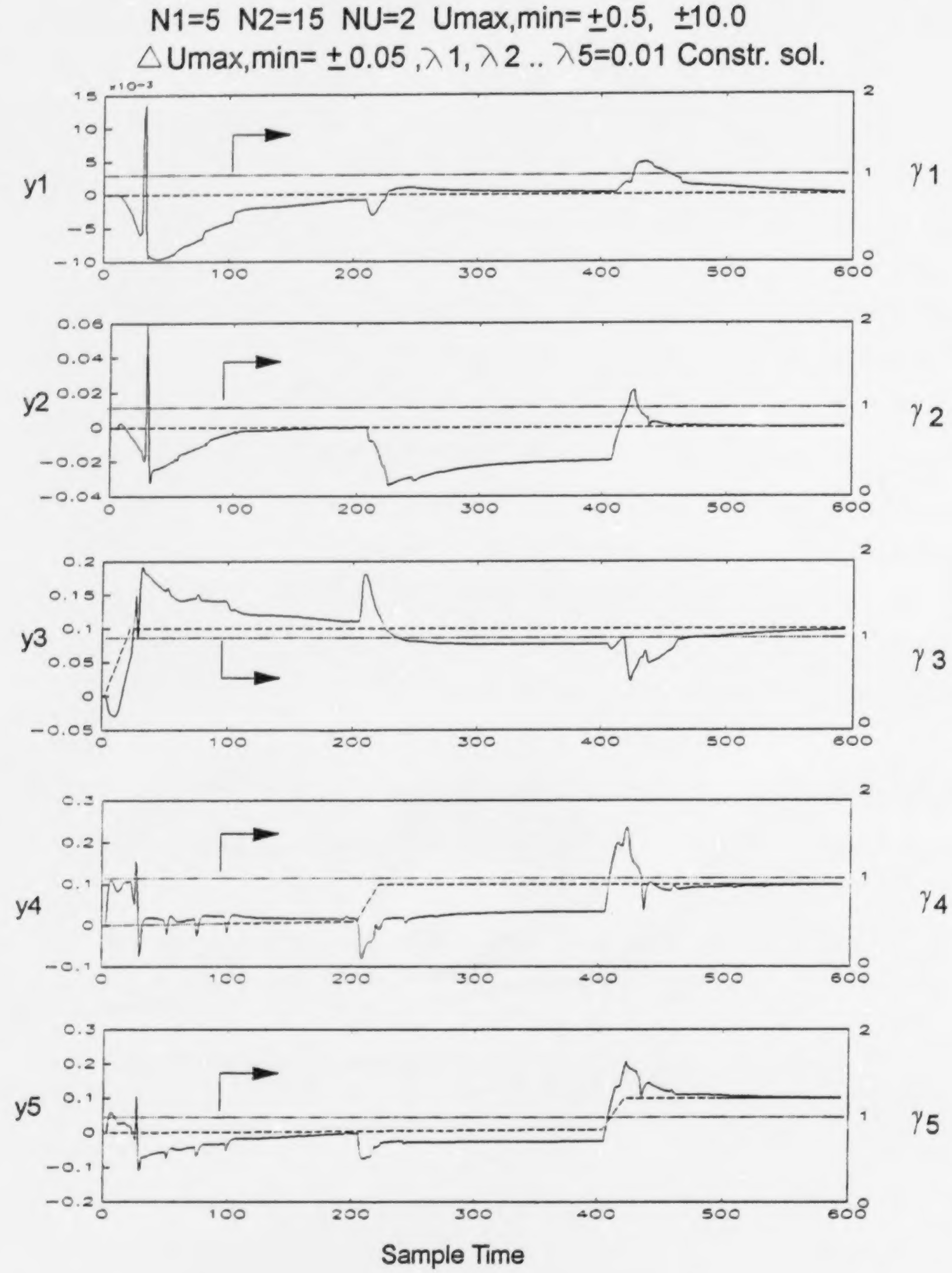


Figure 4.10 Closed loop simulation with tuned T-filter

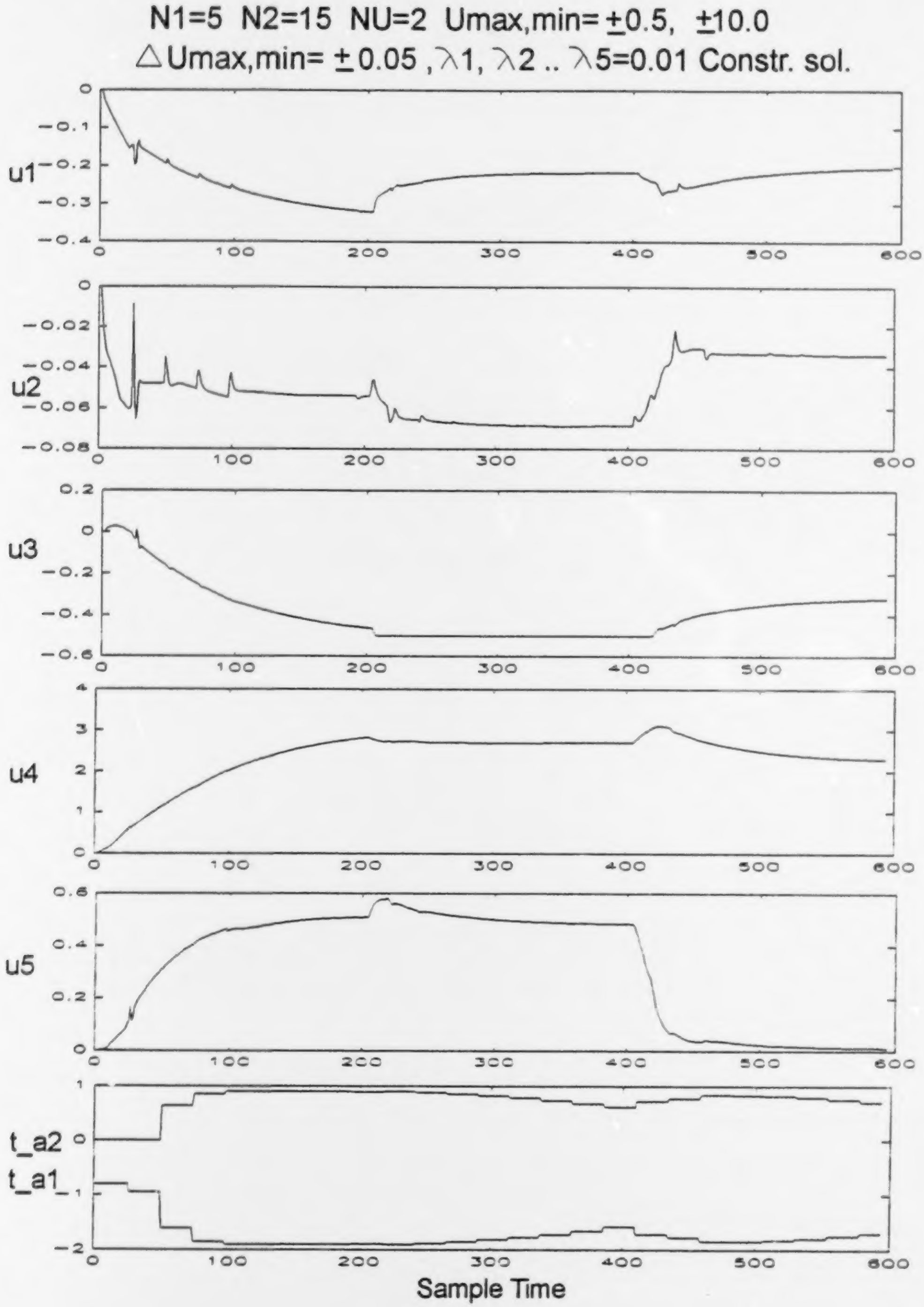


Figure 4.11 Closed loop simulation with tuned T-filter

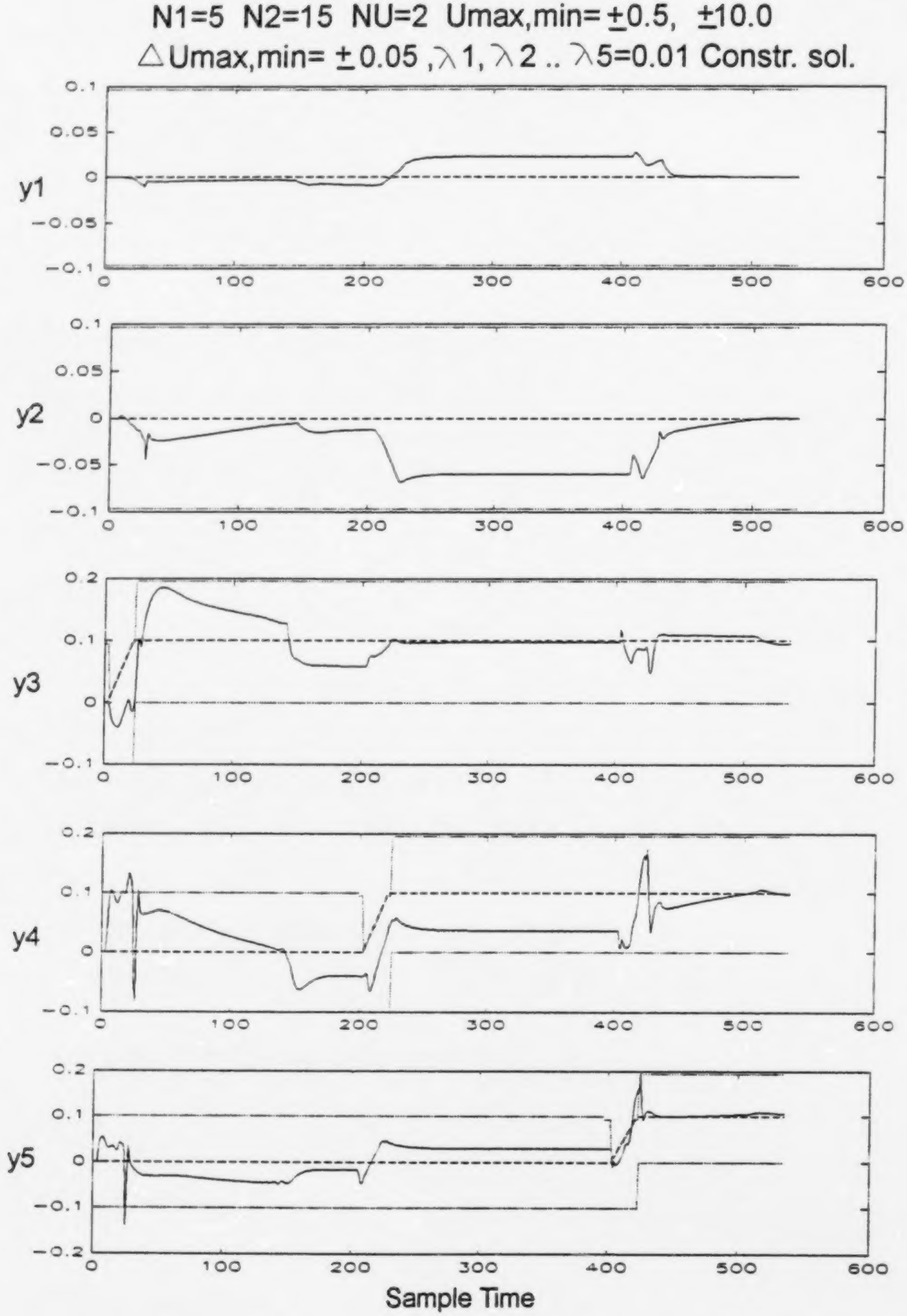


Figure 4.12 Closed loop simulation with output constraints

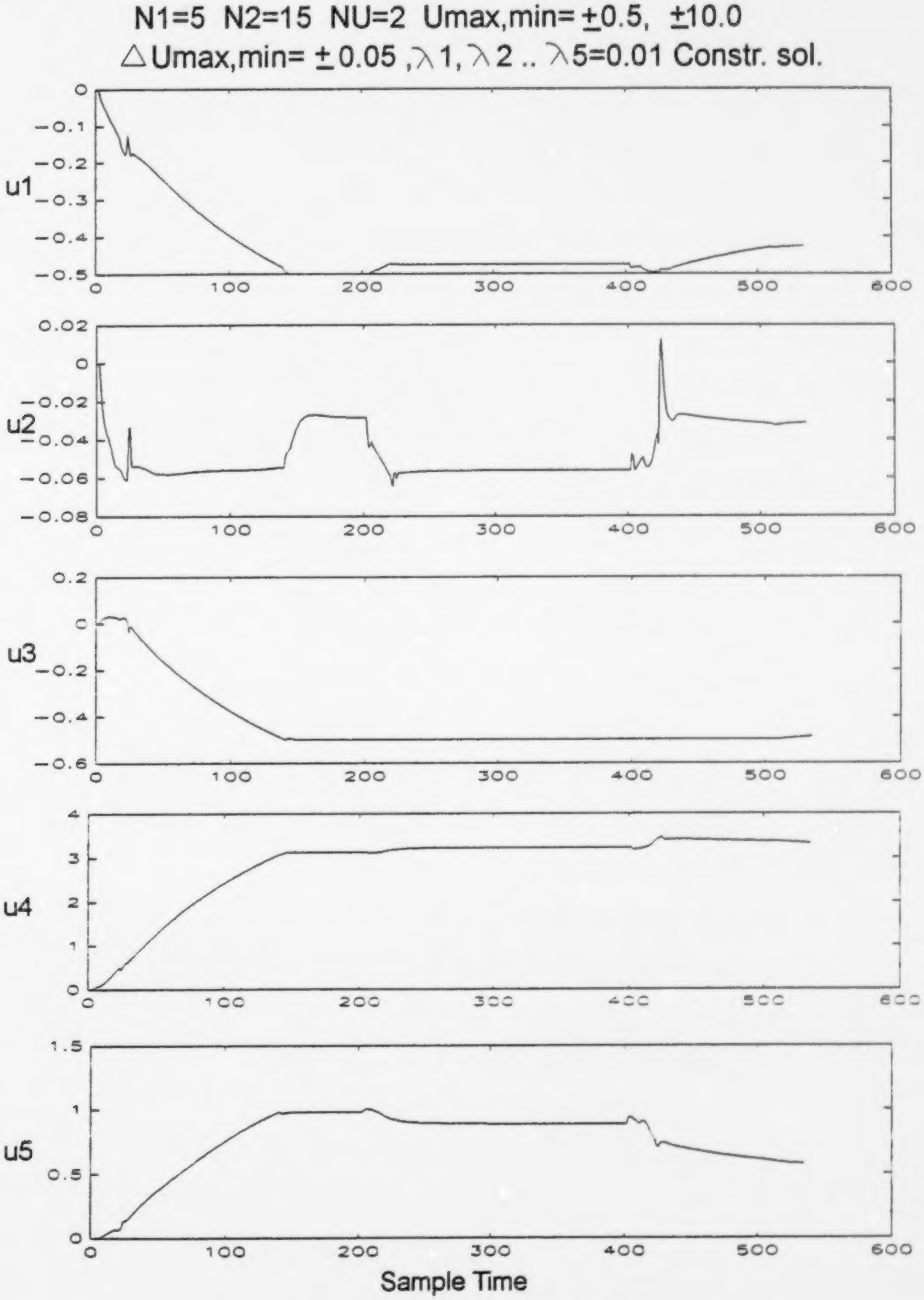


Figure 4.13 Closed loop simulation with output constraints

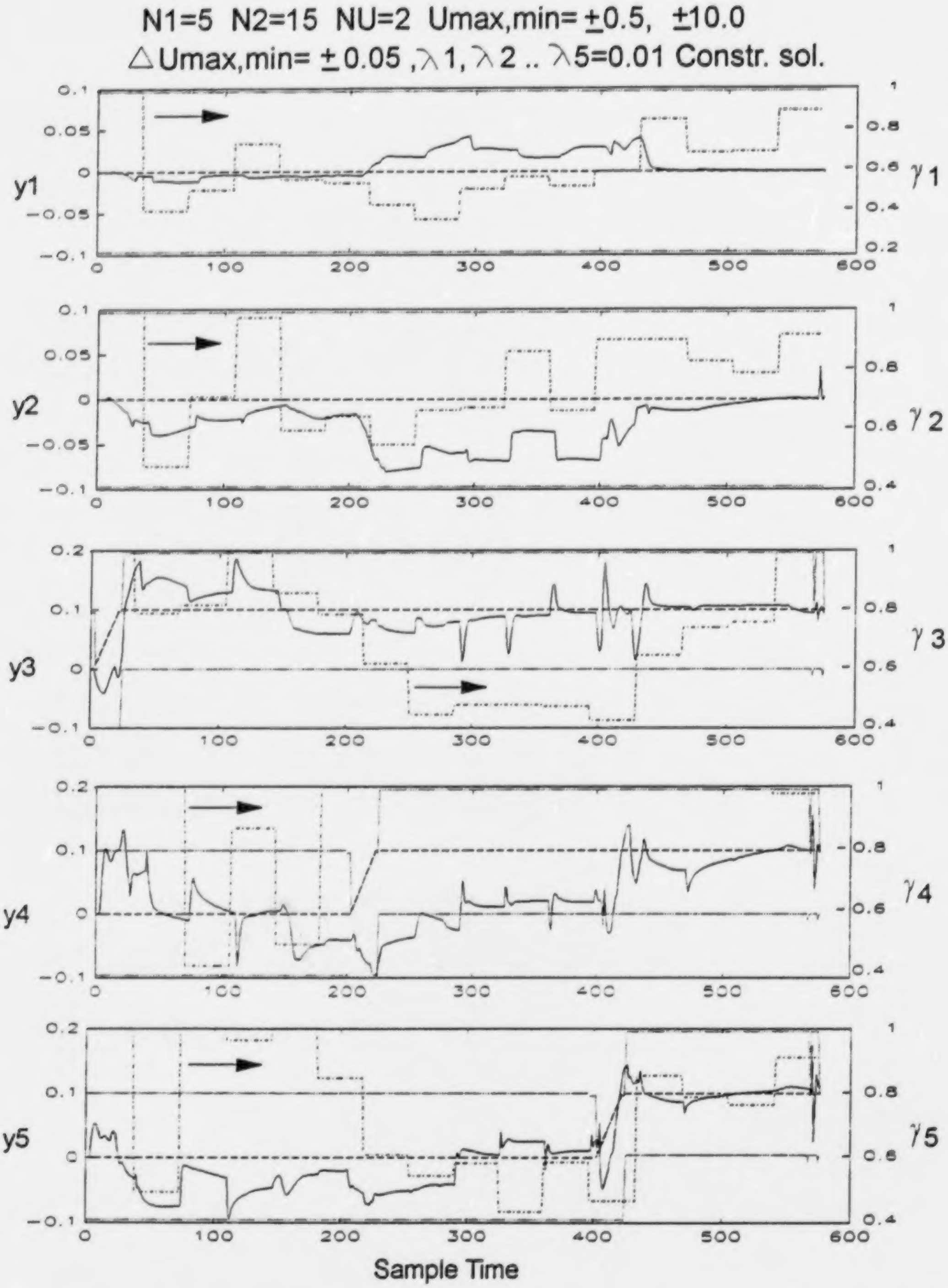


Figure 4.14 Closed loop simulation with output constraints and tuned output weighting

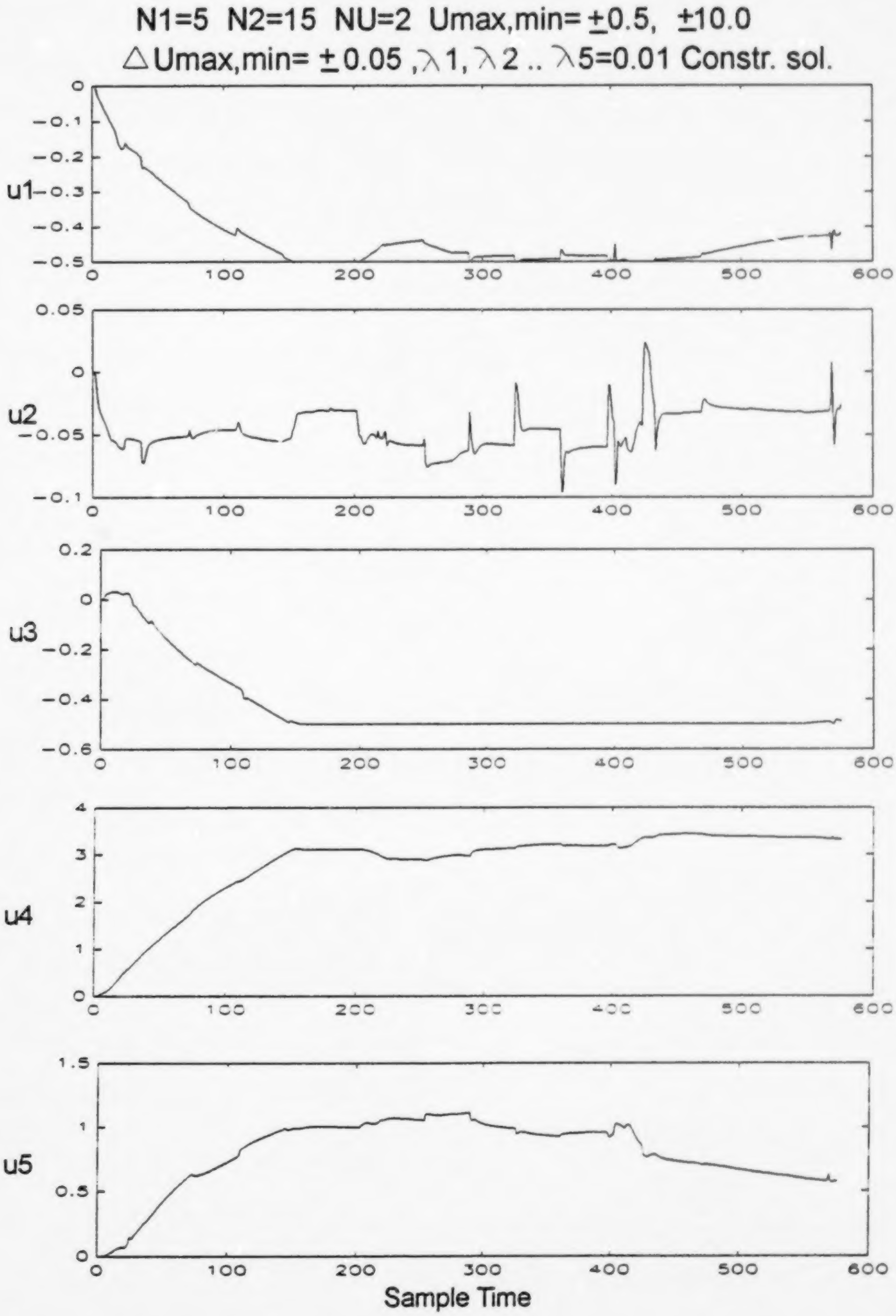


Figure 4.15 Closed loop simulation with output constraints and tuned output weighting

(figures 4.14 and 4.15). Some observations are as follows:

- Comparing figures 4.12 and 4.13 with the base case presented in figures 4.6 and 4.7, we see that the response is identical. There is no improvement in the response when output constraints are present. This is due to the fact that none of the output constraints are violated, thus the solution path generated by QPSOL is the same for the case when the output constraints are present and when the output constraints are not present.
- A more illustrative comparison can be made between figure 4.8 and 4.14. The only difference between the simulation presented in figure 4.8 and figure 4.14 is that output constraints are included in figure 4.14. At sample time 410, the output y4 in figure 4.8 goes above 0.2, which would be a violation of the output constraint. However, looking at y4 in figure 4.14 at sample time 410 shows that it is well within the output constraint of 0.2. There is also a reduction in the overshoot of y5 in figure 4.8 to a much smaller value in figure 4.14.
- The deviation in outputs y1 and y2 is more between sample time 210 to 400 when output constraints are activated (figure 4.14), than when they are inactive (figure 4.8). These deviations, however, do no violate any output constraints, and thus can be taken as 'acceptable' performance. Stringent output constraints can be introduced on outputs which are considered important and need to be controlled within tight bounds.

Output constraints thus enable the user to keep outputs within tight bounds. This idea is very useful in actual processes where safety could be jeopardised by a variable violating its bounds.

4.3.4 Closed loop performance with model plant mismatch

Since GPC is a model based technique, the performance of the controller depends on the model chosen. The 'true' model of the plant can never be determined, thus there is always some mismatch between the model chosen and the 'true' plant. This is referred to as model plant mismatch. The focus of this section is on evaluating the performance of the GPC controller in the presence of model plant mismatch. The mismatch is in the gains of the process transfer functions, as given by table 4.2. The values of ε are as follows:

$\varepsilon_1 = \varepsilon_3 = +0.1$

$\varepsilon_2 = -0.1$

$\varepsilon_4 = -0.2$

$\varepsilon_5 = +0.2$

This translates into a 0.3 to 6.3% mismatch in the gains of *all* the transfer functions. A simulation of the process is presented in figures 4.16 and 4.17. In this simulation, the tuning parameters are the same as decided in the 'selection of initial tuning parameters' section, but both output weighting and the T-filter are tuned simultaneously. From figure 4.16, we see that all the outputs converge to their respective set points at sample time 590. After this time, model plant mismatch is introduced in order to contrast the performance with no model plant mismatch case. Figures 4.16 and 4.17 show a high variance in the outputs and inputs. At sample time 790, set points for outputs y3, y4 and y5 are set to 0.0. The outputs do track the set point but the noise persists and is undesirable.

In order to reduce this high variance in outputs output caused by model plant mismatch, another simulation is carried out, this time with $\lambda_{1,2,3,4,5}$ set to the following values:

$\lambda_1 = 4.0$

$\lambda_2 = 3.0$

$\lambda_3 = 5.0$

$\lambda_4 = 1.0$

$\lambda_5 = 2.0$

These values were not chosen arbitrarily, but were chosen based on the sensitivity of outputs to inputs. The output u4 results in the least amount of movement in outputs y1 to y5 and output u3 results in maximum movement in outputs y1 to y5, thus the λ for u4 is set to 1.0 and the λ for u3 is set to 5.0. The λ's for other outputs were scaled accordingly. The output weighting is kept constant at 1.0 and the T-filter is fixed as $1\text{-}0.8q^{-1}$ until sample time 785 when T-filter tuning is activated. This simulation is presented in figures 4.18 and 4.19. Model plant mismatch is introduced at sample time

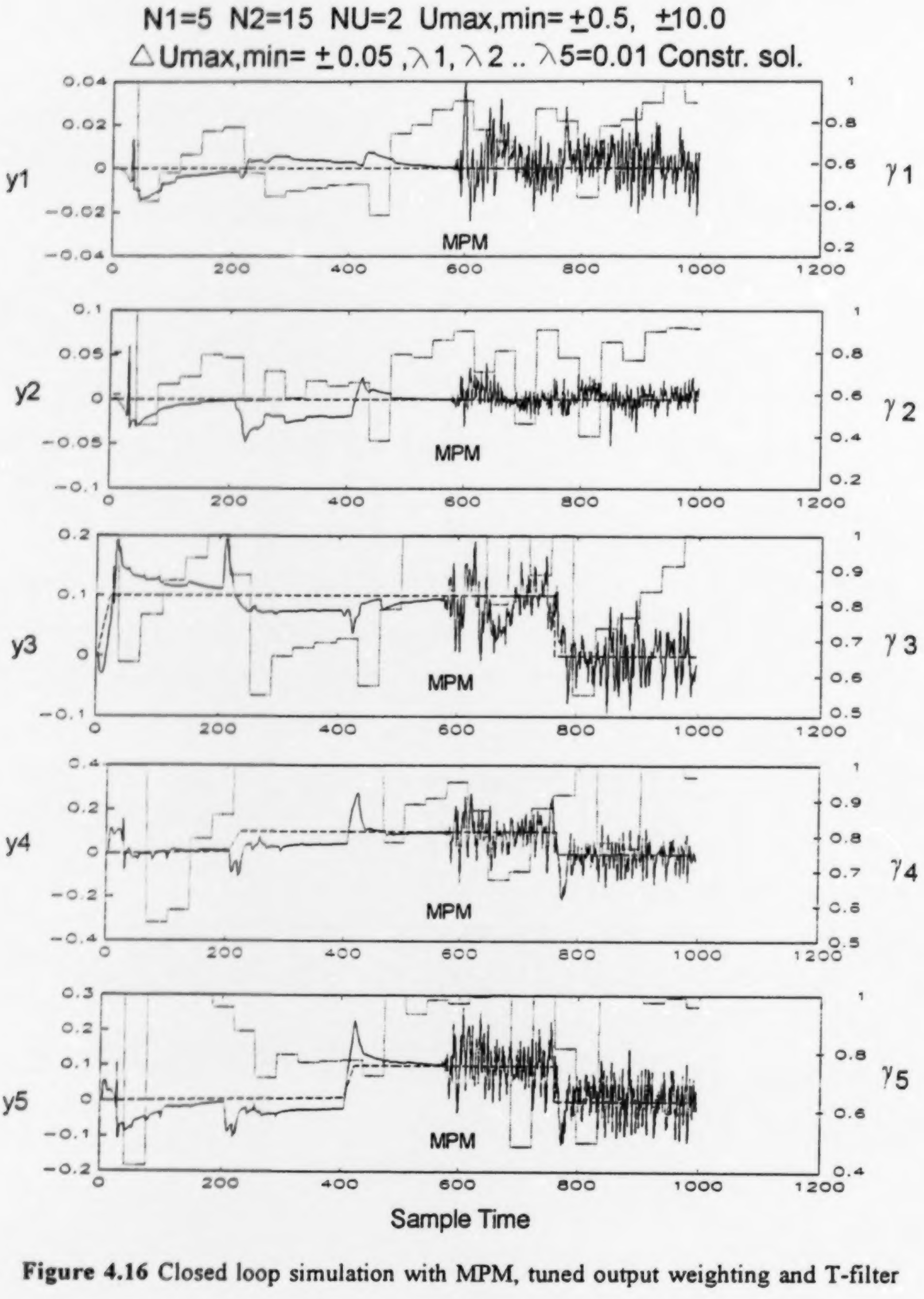


Figure 4.16 Closed loop simulation with MPM, tuned output weighting and T-filter

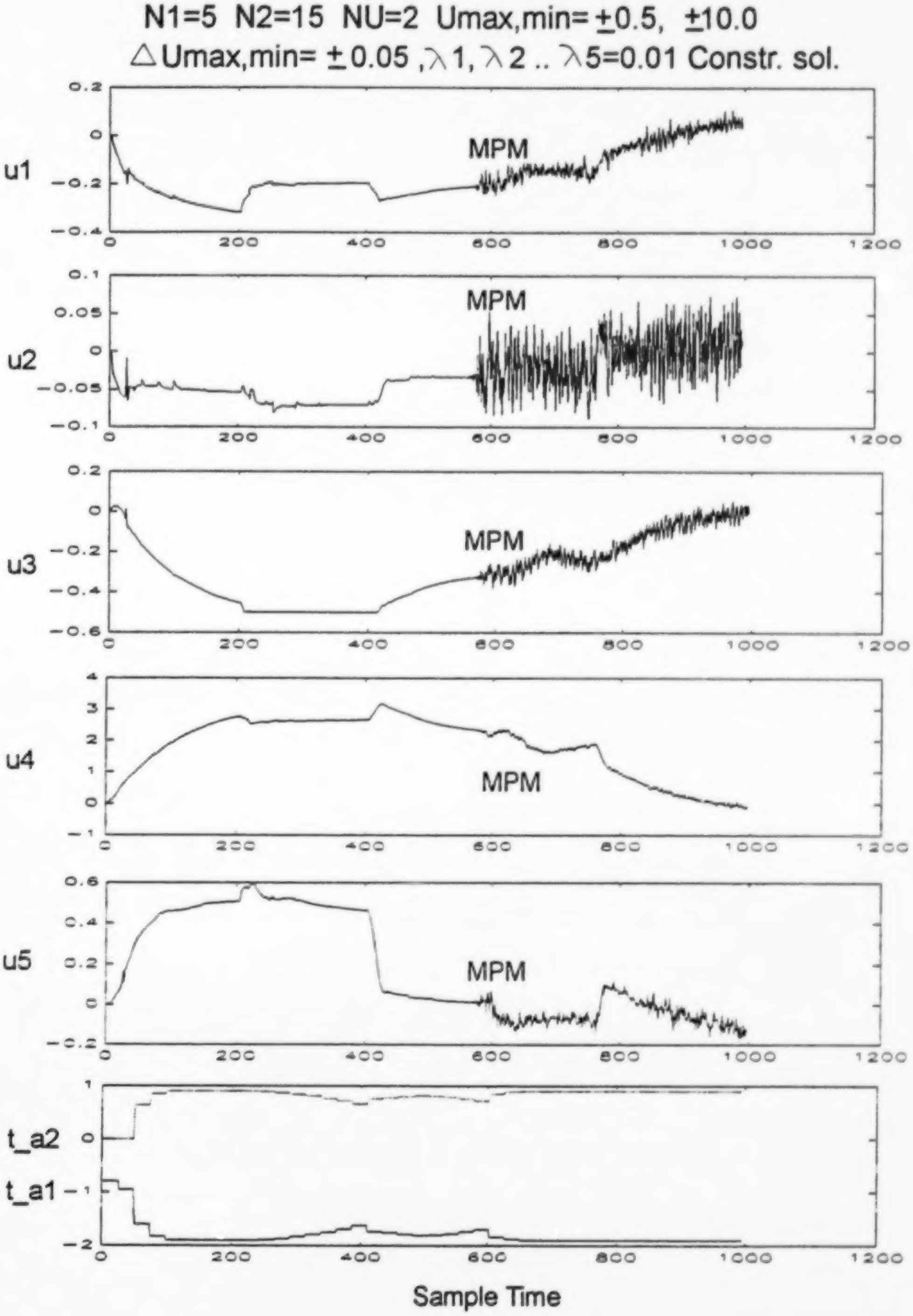


Figure 4.17 Closed loop simulation with MPM, tuned output weighting and T-filter

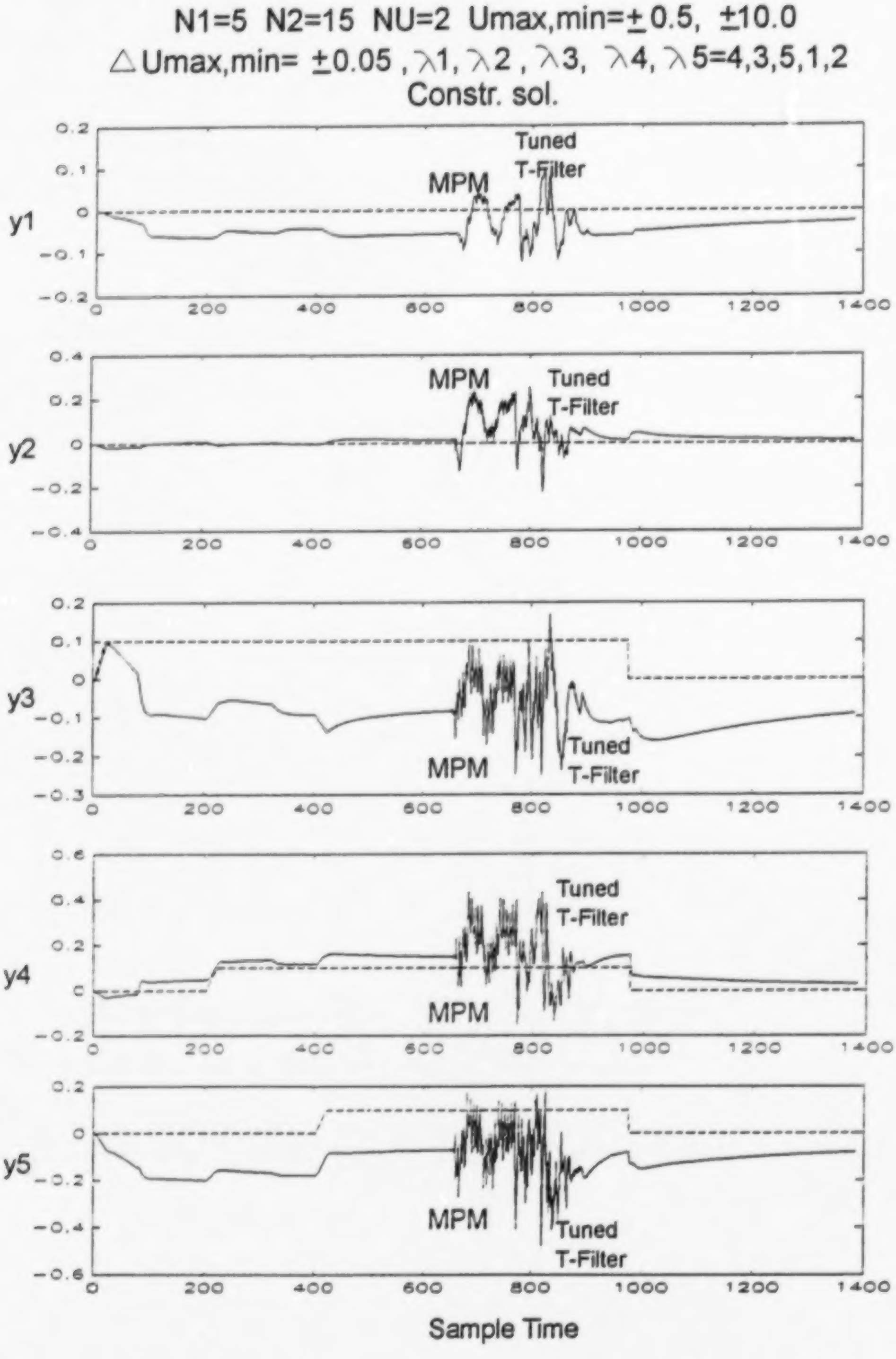

N1=5 N2=15 NU=2 Umax,min=± 0.5, ±10.0
△Umax,min= ±0.05 , λ1, λ2 , λ3, λ4, λ5=4,3,5,1,2
Constr. sol.

Figure 4.18 Closed loop simulation with MPM and tuned T-filter

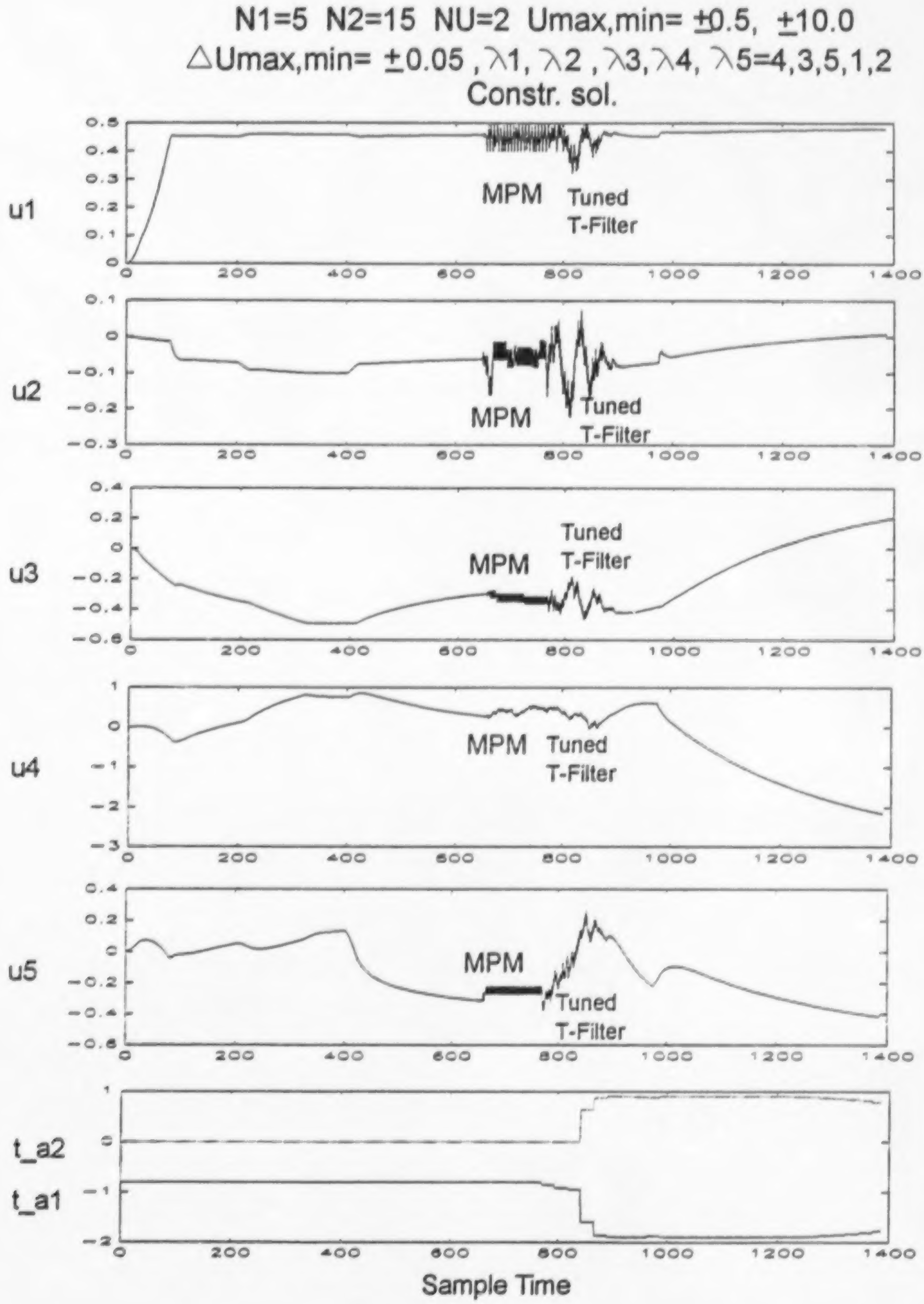


Figure 4.19 Closed loop simulation with MPM and tuned T-filter

610. The following can be said about this simulation :

- Before model plant mismatch is introduced at sample time 610, the outputs are converging to the set points very slowly. The offset between the outputs and the set points is more than the base case (figures 4.6 and 4.7). This is understandable because the input or λ weighting is two orders of magnitude higher than the base case, resulting in very little movement of inputs u1, u2, u3, u4 and u5, consequently slowing down the outputs.
- After model plant mismatch is introduced at sample time 610, the high variance seen in figures 4.16 and 4.17 appears. At sample time 770, T-filter tuning is started and by sample time 840, a second order filter is implemented, reducing the variance. After sample time 900, the high variance is eliminated by the implementation of a second order T-filter and the outputs begin converging to their respective set points.

This clearly illustrates the ability of the T-filter to reduce high variances in outputs of a process and make the process robust to model plant mismatch. Also, λ weighting helps in dealing with model plant mismatch by reducing the movement in inputs u1, u2, u3, u4 and u5. This idea of using λ weighting to eliminate noise due to model plant mismatch is apparent from the theory of GPC. The reader is referred to section 2.2.2 in which the objective function of GPC is discussed.

4.3.5 Simulation of a non-square system

There are many processes in industry which do not have equal number of inputs and outputs. When the number of inputs is less than the number of outputs, then the system is *underspecified* and is not controllable. However, when there are more inputs than outputs, then not only is the system controllable, but there exist extra degree(s) of freedom. This *overspecified* system has the following implications:

- The extra degree(s) of freedom can be ignored and not used at all in the control system, making it a square system.
- If an *overspecified* plant were operated manually, then there can be more than one set of values of the inputs for the same operating point. If there is a cost ($)

associated with each input, then an informed choice of the values of inputs can reduce operating costs.

- In some situations, extra degree(s) of freedom can be reserved to be used in case of the failure, saturation or maintenance of regularly used inputs. In the case of constrained GPC, the QPSOL algorithm will use the extra degree(s) of freedom in an "optimal" sense and calculate a control trajectory with the minimum cost, J (see section 2.2.2 for definition of the cost function).

A preliminary investigation is carried out to explore the possibilities of using expert systems to resolve some of the above mentioned issues.

Specifically, simulations with a 5-input 4-output process are performed to demonstrate that the final values of inputs achieved are different for the same operating point when:

1. all the 5 inputs are used;
2. only 4 of the 5 inputs are used and
3. the extra degree of freedom is used when one of the original inputs saturates.

The process model used is the same as for the 5-input 5-output simulations (the Shell Control Problem), without output y5. These simulations are presented in figures 4.18 to 4.23. In all the simulations, a set point change is made in output y4 (from 0 to 0.2), while all other set points stay at 0.0. Other GPC tuning variables are as follows:

$N1 = 5$

$N2 = 15$

$NU = 2$

$U_{max,min,1,2,3} = \pm 0.5$; $U_{max,min,4,5} = \pm 10.0$

$\Delta U_{max,min} = 0.05$

$\lambda_i = 0.01$ $i = 1,2,3,4,5$

The following observations can be made from these figures:

- When all the 5 inputs are used, (figures 4.18 and 4.19), the convergence to set points is fast after the initial interactions. Also none of the inputs saturate at their amplitude constraints. The approximate final values of the outputs are:

  u1 = 0.3, u2 = -0.01, u3 = 0.07, u4 = -0.5 and u5 = -0.7

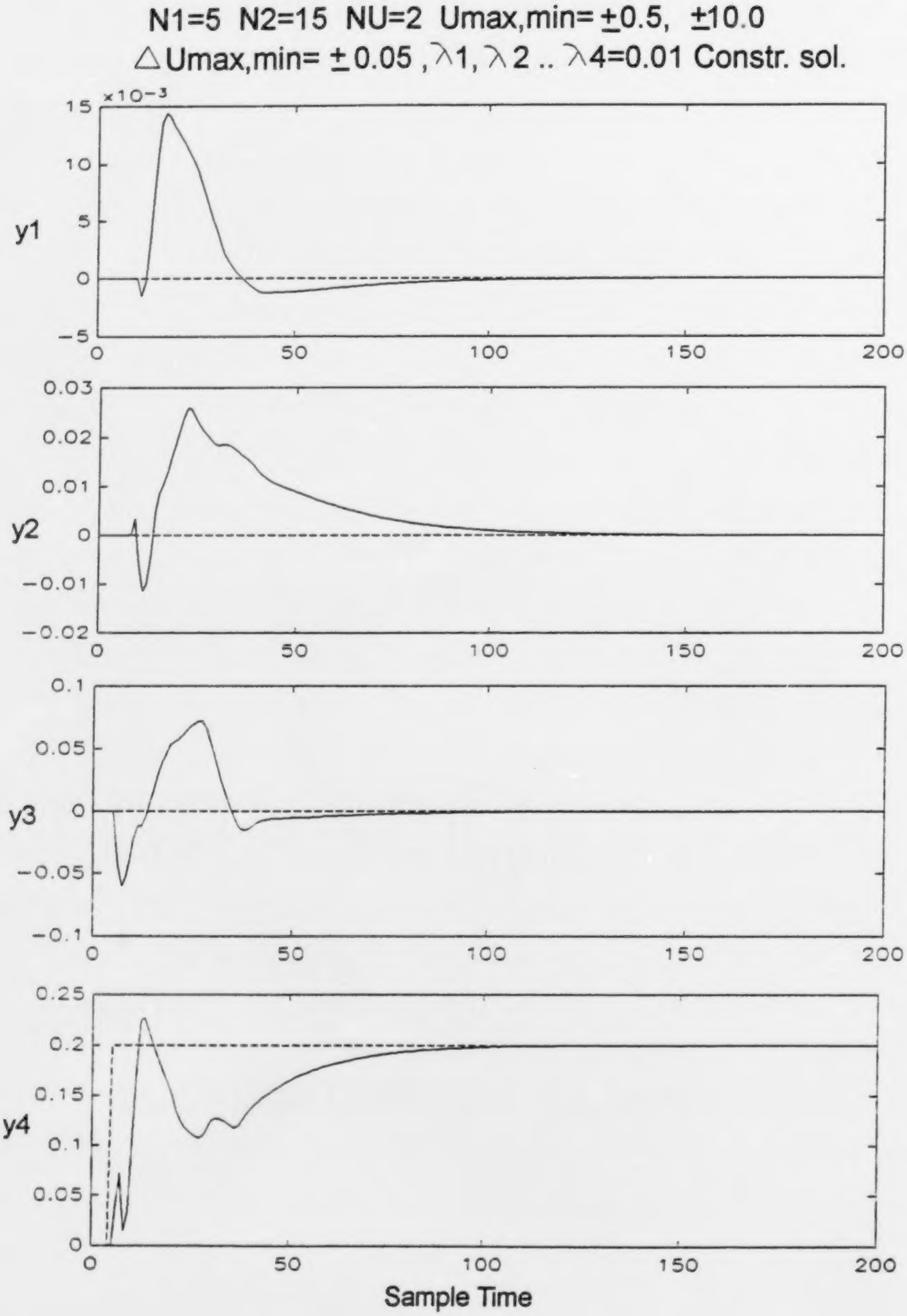


Figure 4.20 Simulation of a 5-input 4-output system with all 5 inputs active

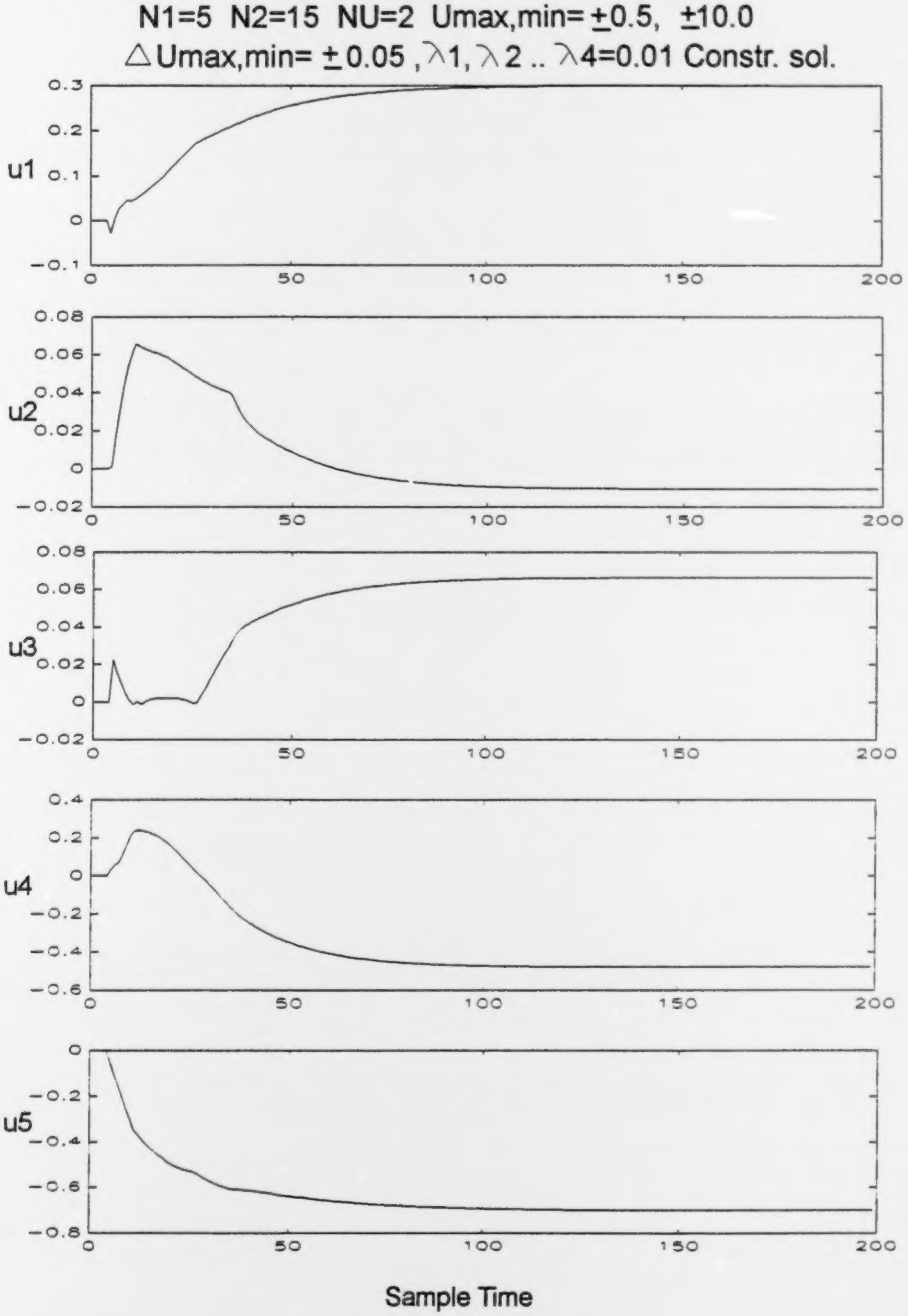


Figure 4.21 Simulation of a 5-input 4-output system with all 5 inputs active

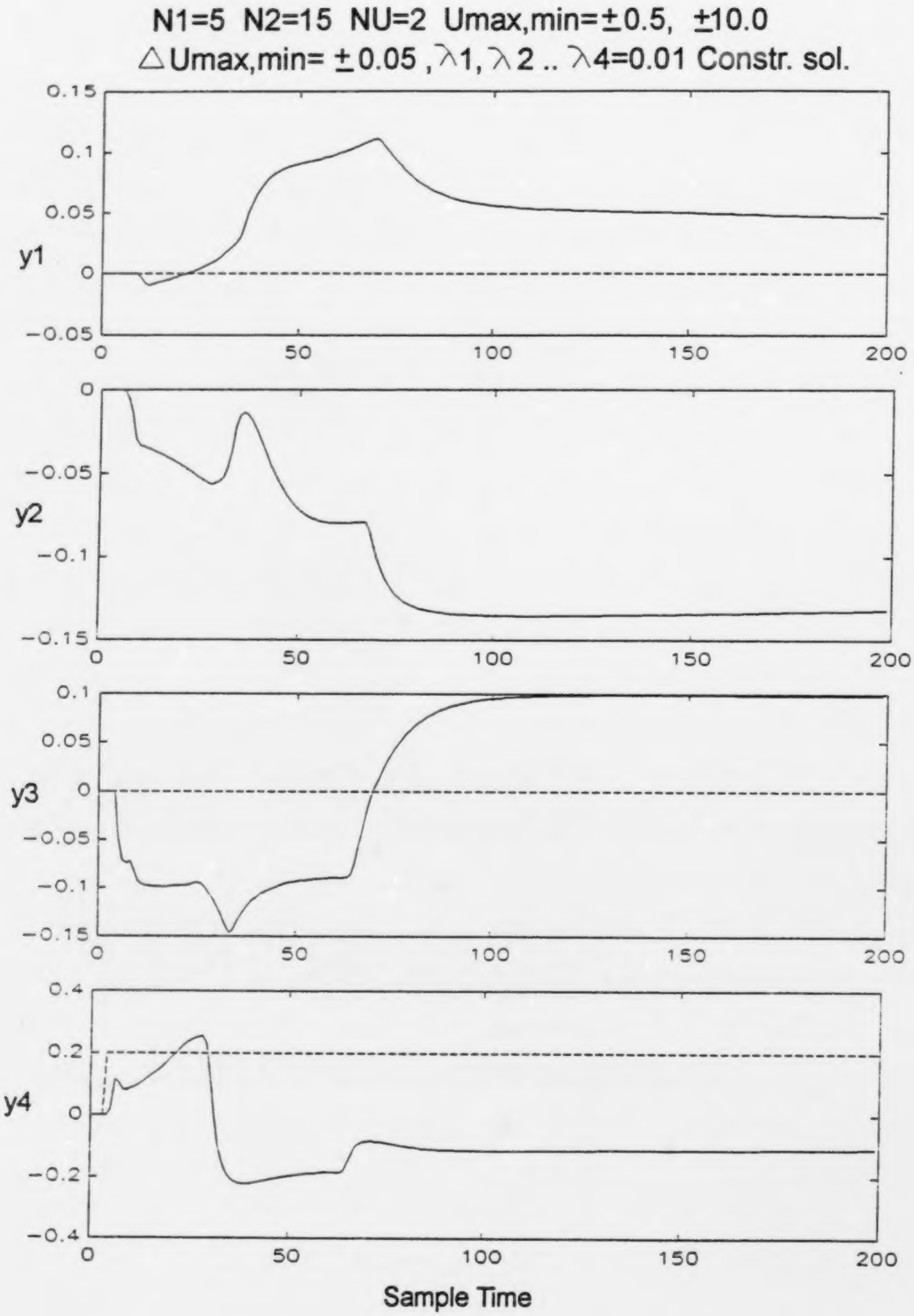


Figure 4.22 Simulation of a 5-input 4-output system with 4 inputs active

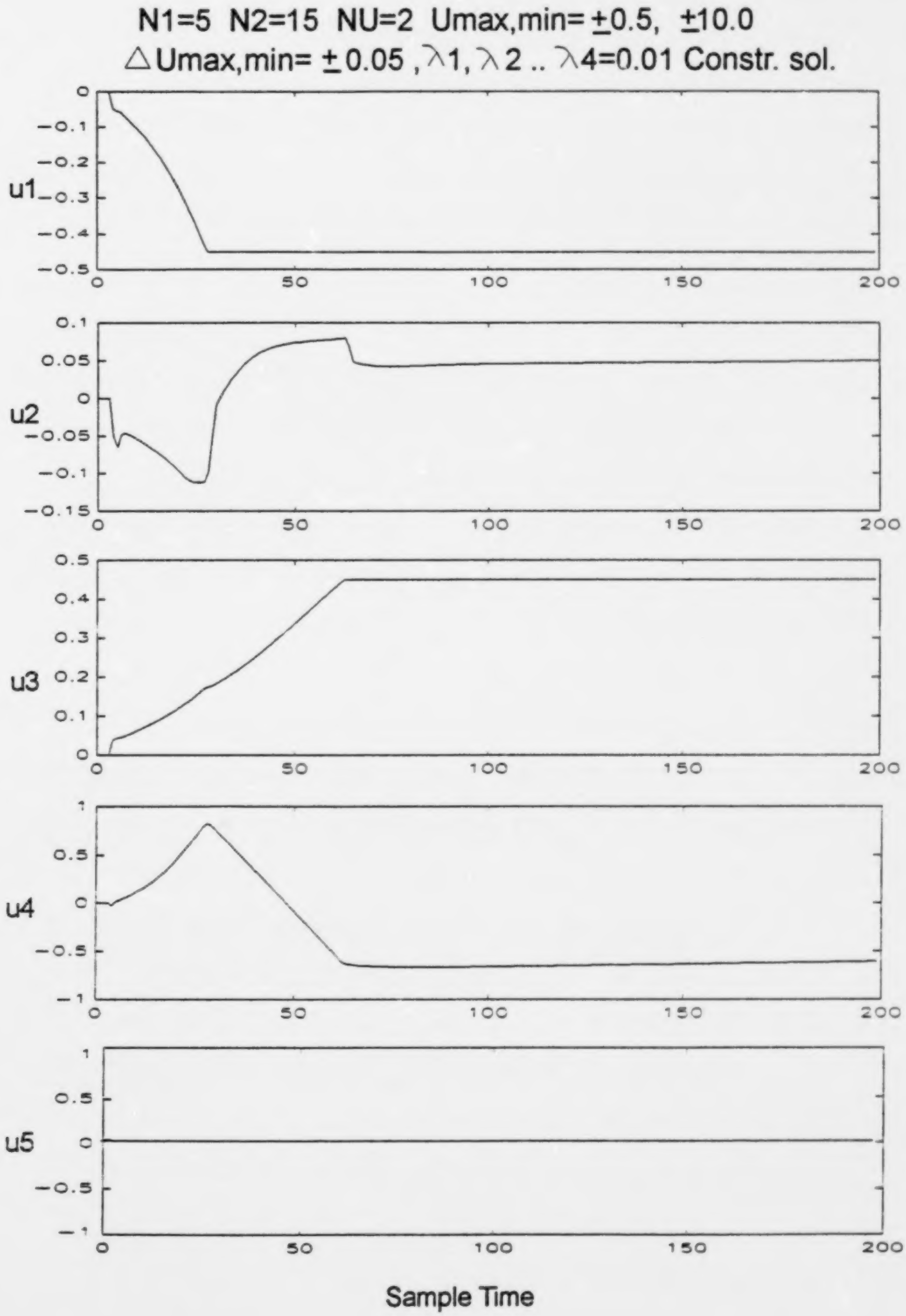


Figure 4.23 Simulation of a 5-input 4-output system with 4 inputs active

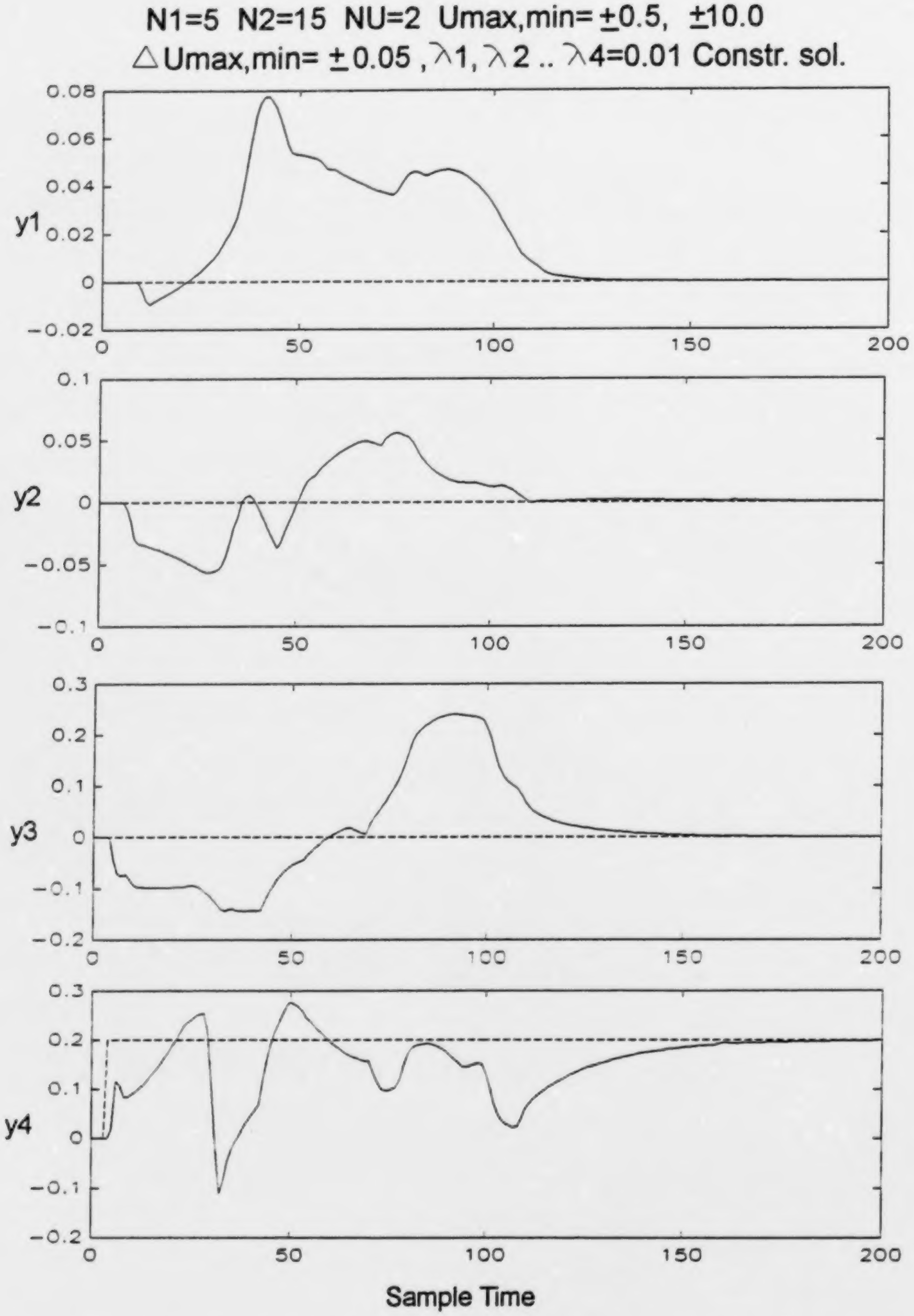


Figure 4.24 Simulation of a 5-input 4-output system with saturation detection

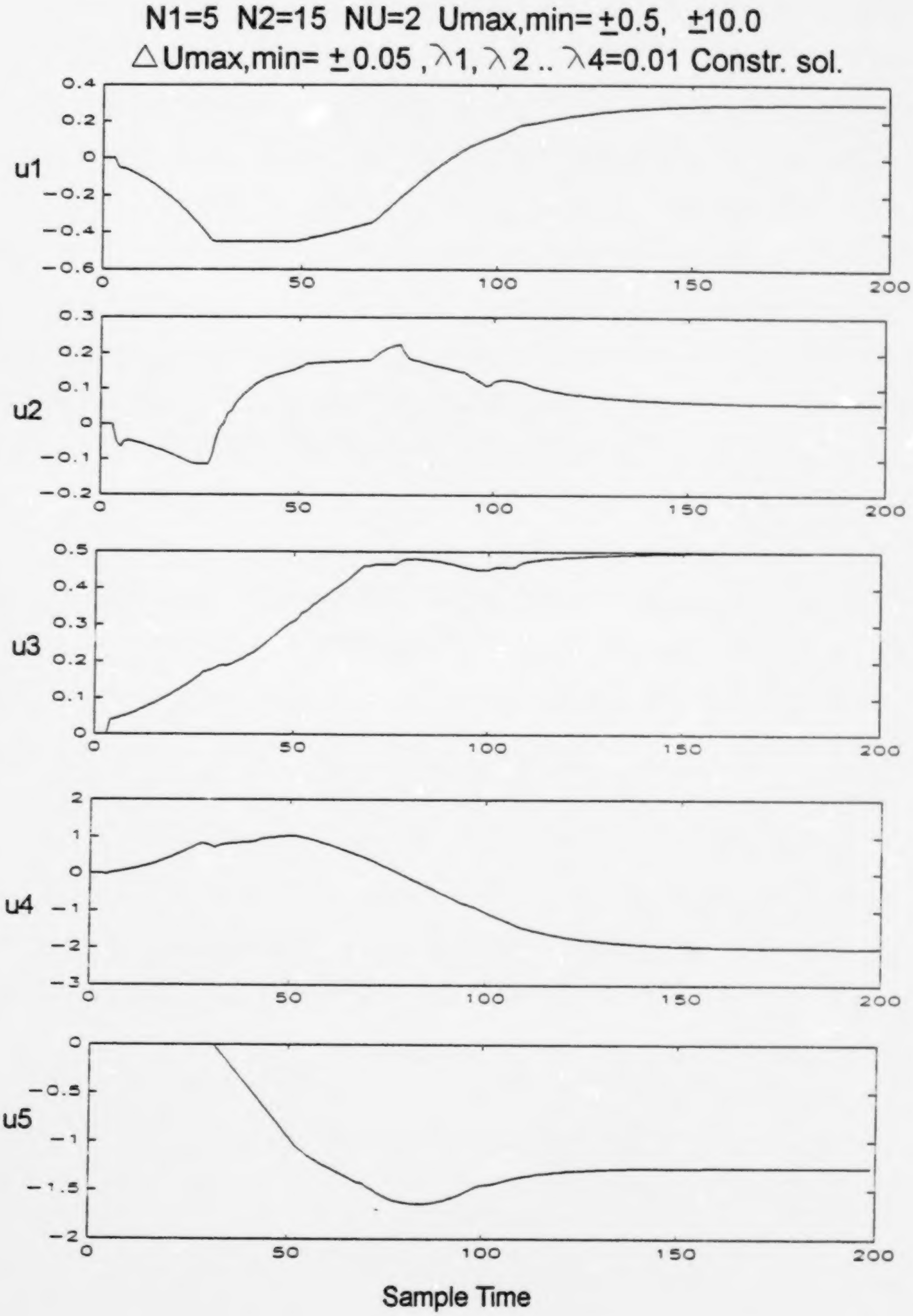


Figure 4.25 Simulation of a 5-input 4-output system with saturation detection

- Figures 4.20 and 4.21 show results of simulations when 4 of the 5 available inputs are used. The inputs u1 and u3 saturate at times t=25 and t=60 respectively, causing offset at steady state in all the outputs. This means that the set point is infeasible when 4 inputs are used.
- Finally, consider the simulation in figures 4.22 and 4.23. In this simulation, input u5 is not used until one of the original inputs u1, u2, u3 or u4 saturates. Figure 4.23 shows that input u1 saturates at time t = 30 and immediately input u5 is automatically activated. All outputs converge to their respective setpoints by time t=180. The approximate final values of the outputs are:

u1 = 0.3, u2 = 0.05, u3 = 0.5, u4 = -2.0, u5 = -1.3

The final values of inputs in figure 4.23 are completely different from those obtained from the simulation in figure 4.19, although the process operating point and all the tuning parameters are the same. This clearly implies that the solution of the constrained GPC problem is non-unique for *overspecified* systems.

Practically, this idea of a non-unique solution can be very important if some inputs are more expensive than others. An expensive input (in this case u5) can thus be reserved to be used only when required.

In order to implement this idea of using an input only when required, rules were added to the ALPS knowledge. These rules do essentially two things:

1. Detect when an input saturates at its amplitude constraint or when it moves from saturation back into the feasible region.
2. Relax the high value of the input weighting, λ, on the extra degree of freedom (u5) when one of the inputs saturates, or implement a high value of λ when an input moves from saturation back into the feasible region.

A listing of rules which do the above are in Appendix B.

4.4 Conclusions

This chapter deals with the Shell benchmark problem and discusses the design of a controller tuning strategy which automatically updates the output weighting and the T-filter. The output weighting is tuned such that the channel which deviates from the set point the most gets the maximum output weighting of 1.0, while the other channels get scaled automatically relative to the channel with the maximum deviation. The T-filter is tuned based on the variance of the inputs. A first or second order T-filter can be implemented by the autotuner, depending on the variance of the inputs. Selection of *initial* tuning parameters is critical and a set of *rules* was developed for choosing them. These rules were validated by simulations.

The effect of the autotuner on closed loop response is also evaluated. First the effect of output weighting tuning is determined and then the effect of T-filter tuning is evaluated. In both cases, the closed loop response improved. The T-filter tuning is especially useful in scenarios where model plant mismatch is present. Output constrained GPC is also evaluated. The simulations show that the presence of output constraints reduce large overshoot in the outputs. The combination of output constraints and tuned output weighting is quite powerful in maintaining outputs within their constraints.

The effect of model plant mismatch on closed loop performance is also evaluated. With appropriate values of the input weighting and T-filter, the high variance in outputs due to model plant mismatch is eliminated.

Simulations of a 5-input 4-output, non-square system show that inputs can take different final values for the same operating point.

In addition to the simulations presented in this chapter, a number of other simulations were performed. Based on the experience of the author and the knowledge of theoretical results, a table with comments is drawn. This table is presented on the following page. It should be noted that some of the comments may be applicable for this process only, since it is unique in its complexity. However, a theoretical explanation of why some of the simulations worked and some did not, is attempted.

Run #	N1, N2, NU	$\lambda_{1,2,3,4,5}$	$\gamma_{1,2,3,4,5}$	MPM ?	Constr. ?	T-Filter tuned ?	γ-wtg. tuned ?	Comments	Supported by Theory ?
1	1, 15, 1	0.01, ...0.01	1.0, ...1.0	N	Y	N	N	There is offset at steady state due to the inputs saturating at their constraints. The Unconstrained case is shown to be unstable.	Yes. There is offset due to the decrease in the number of degrees of freedom. Constraints help in stabilising an unstable system.
2	5, 15, 1	0.01, ...0.01	1.0, ...1.0	N	Y	N	N	same as above	same as above
3	1, 15, 2	0.01, ...0.01	1.0, ...1.0	N	Y	N	N	same as above	same as above
4	5, 15, 2	10.0, ...10.0	1.0, ...1.0	Y	Y	N 2nd ord. filter	N	The response for run # 4 & 5 is the same. In both cases, there is offset at steady state due to saturated inputs. The response is also a lot slower. Model plant mismatch can be handled as well, provided a 2nd order filter is used.	The idea of using λ weighting or rate constraints interchangeably is supported by theory since both methods make the solution of the constrained GPC problem more conservative.
5	5, 15, 2	0.01, ...0.01	1.0, ...1.0	Y	Y Rate Cons =0.005	N 2nd ord. filter	N		
6	5, 15, 2	4,3,5,1,2	1,1,4,8,8	Y	Y	N 2nd ord. filter	N	The different values of γ weighting reduce interactions as compared with run#4. There is overall improvement in the response. The effect of increasing λ is the same as for run#4.	Since weighting the outputs differently reflects in the objective function, the solution of the QP problem is such that it moves the output with maximum weighting the most, resulting in a decoupling effect.
7	5, 15, 2	0.01, ...0.01	1.0, ...1.0	Y	Y Rate Cons =0.04	Y	Y	In this case the t-filter and the output weighting are tuned simultaneously. Due to a smaller rate constraint in run#8, we find that the process response is more sensitive to output weighting tuning. but less sensitive to model plant mismatch. The higher rate constraint in run#7 results in a process response less sensitive to output weighting tuning but very sensitive to error due to model plant mismatch.	This can be explained by considering the objective function and the solution methodology of GPC. The higher rate constraint in run#7 results in a more aggressive control action, resulting in a 'noisy' output and the masking of any sensitivity to output weighting. For the case when the rate constraint is 0.01, conservative control action is calculated, resulting in smoother but slower control. Sensitivity to output weighting is due to the larger error because of the slower response and cumulative error in the objective function being dominated by the terms due to output weighting.
8	5, 15, 2	0.01, ...0.01	1.0, ...1.0	Y	Y Rate Cons =0.01	Y	Y		

Note: 1. Default t-filter is 1st order unless otherwise specified.

2. Default constraints are: Umax,min (1,2,3) = +- 0.5

Umax,min (4,5) = +- 20.0

Du max,min = +- 0.05 (Rate const.)

3. All responses are to set point changes only.

Chapter 5

Experimental evaluation of autotuning GPC

The ultimate goal of doing simulations and controller design is to be able to test the design on a real process. It is thus important to give careful thought to the choice of programming language(s) used and the portability of the development to other applications on different hardware systems. Most of these considerations are taken care of in this project. All the programming is in ANSI standard 'C' and FORTRAN. The knowledge base development tool, Gensym's G2, is used for developing the tuning rules. The OPTO-22 process interface is used to send and receive data from the process. Thus this application can be executed on any computer which has 'C' and FORTRAN compilers, the G2 software and a hardware link with an OPTO-22 process interface.

In the process of implementing autotuning GPC on the experimental setup there were a few 'stumbling blocks', each of which resulted in valuable learning experience and an appreciation for the amount of work involved in making an application real-time and on-line. This experience was valuable in understanding the importance of testing communications and data integrity, as well as learning about process instrumentation and the OPTO-22 process interface.

This chapter is organized into the following sections: section 5.1 discusses the pilot process on which this application is implemented. The model of the process is discussed in section 5.2 and the importance of choosing the correct type of model is also discussed. Developing software for the interface between the computer and the process is the focus of section 5.3. Some experimental results are presented in section 5.4. Conclusions are presented in section 5.5.

5.1 Process description

The pilot scale process considered for the application of autotuning GPC is a 2-input 2-output process. The objective of the process is to control the temperature and level of water in a tank by manipulating the cold and hot water inlet flow rates. Some details of the OPTO-22 process interface are also discussed in this section.

5.1.1 The Continuous Stirred Tank Heater

A schematic of the 'Continuous Stirred Tank Heater', hereafter referred to as the CSTH, is presented in figure 5.1. The CSTH consists of a glass tank 14.5 cm. in diameter. The glass tank has a steam coil going through it and contains a baffle and a stirrer. The purpose of the steam coil is to introduce disturbances in the temperature, without changing the level. The stirrer mixes the water in the tank so that there is uniformity in the temperature readings. The baffle is used to reduce noise in the level readings since introducing the water directly into the tank causes splashing and consequently noise in the level readings.

There are a number of measurement transducers to record temperature and one transducer to record level. The temperature transmitters are placed at regular intervals along the length of a long pipe through which the water exits after mixing in the tank. The reason for placing these transducers at different positions is to introduce a measurement delay in temperature. The output of all these temperature transducers or thermocouples goes to a selector switch, from which the desired thermocouple can be selected. The level transmitter is a differential pressure device and can be calibrated to a desired range of measurements. The output of the thermocouple selector and the level transmitter goes to the OPTO-22 process interface. The output of the OPTO-22 goes to the hot and cold water valves. When a disturbance in temperature only is required, the

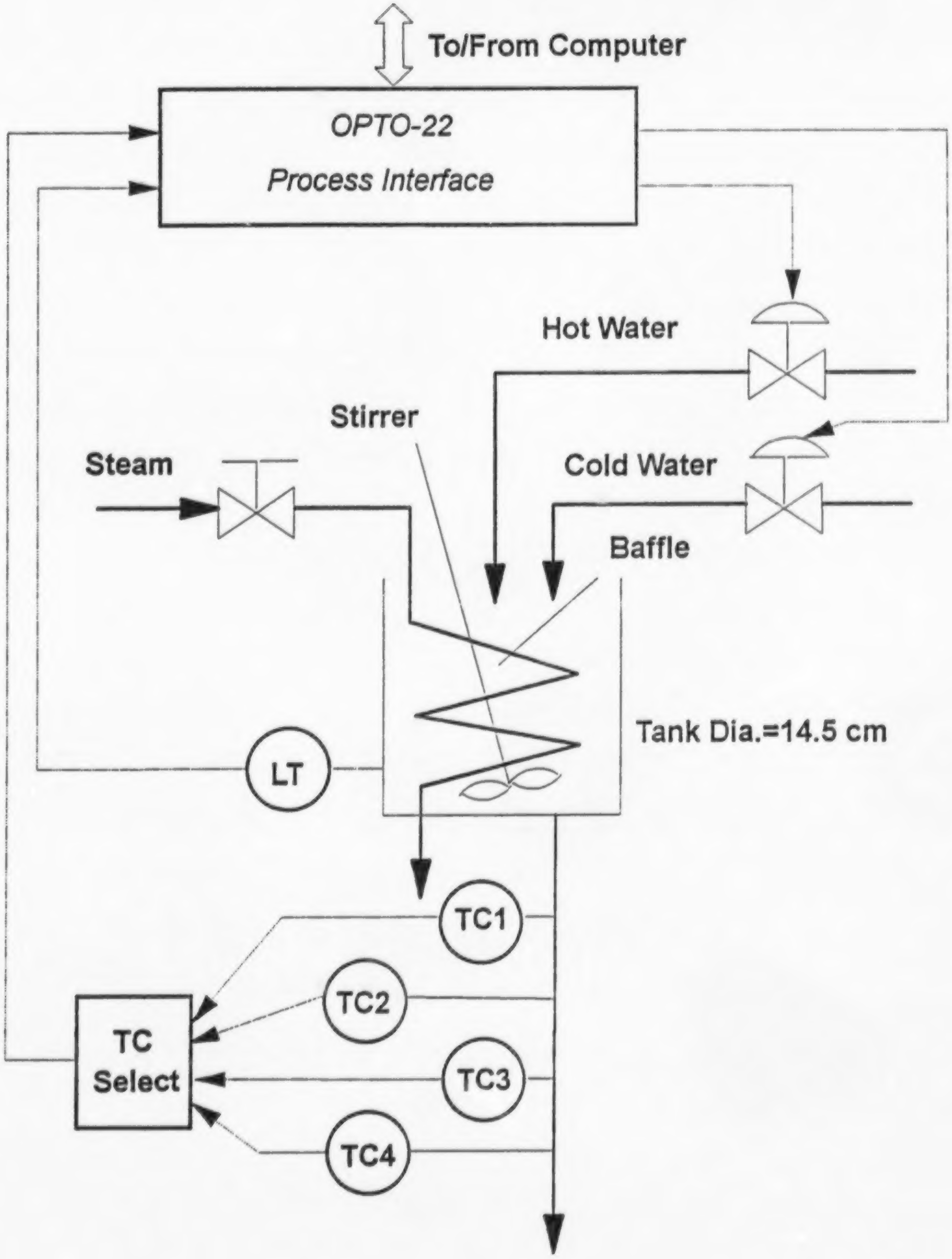


Figure 5.1 Schematic of the Continuous Stirred Tank Heater

steam valve is turned on manually for about 20 seconds. A disturbance in the level only can be made by regulating the exiting water stream from the tank. Disturbances in the temperature and level can be produced by regulating the cold or hot water streams.

5.1.2 The OPTO-22 process interface

The function of the OPTO-22 process interface is to convert data on the RS-232 line from the computer to a signal recognizable by the valves and to convert the signal from the temperature and level transducers to data, and to send it to the computer via the RS-232 cable about 80 metres long. An illustration of the hardware configuration of the process is presented in figure 5.2. The RS-232 is a standard for connecting two pieces of equipment via a serial port. The computer is on the 4th floor, while the process is on the 2nd floor. The reason for using this configuration is because the G2 software is available on the HP 9000 computer, which is set up on the 4th floor and the computer is not portable. The serial port is configured as follows:

- The baud rate is set to 9600 baud.
- The data word contains 10 bits as follows:
  - 1 start bit
  - 8 data bits (no parity)
  - 1 stop bit

The OPTO-22 interface works like a small computer. It has the ability to read/write 16 variables concurrently. Each variable on the process is associated with a 'channel' on the OPTO-22. Each 'channel' on the OPTO-22 can either be configured to read a variable, for example level of the CSTH equipment, or write to an actuator or valve. The 'channel' can also be set up to either operate in a range of 0-5 V or 4-20 mA.The OPTO-22 takes commands in a specific format and puts out commands in a specific format as well. The command is broken up into 3 parts:

- **Beginning:** This part of the command always contains the start of message character, which is ' > ' and two hexadecimal characters representing the address of the OPTO-22. An example of this could be ' >FF '. The ' FF ' stands for the address of the OPTO-22. It should be noted that each OPTO-22 has a specific address. This is important when communicating with a bank of OPTO-22's.

- **Middle:** This part of the command contains 1 to 4 fields as shown below. The fields shown in brackets are optional.

  cmd [position] [modifier] [data]

  cmd: This field always contains a single character which specifies the command to be executed.

  position: This field is required by some commands to specify the 'channels' to be accessed.

  modifier: This one character field is required by some commands to further specify command execution (*i.e.* what type of time delay to set). This option is not used in the present application.

  data: This field is used when we need to write a certain value to a module on the OPTO-22. The value written ranges from 000 to FFF in hexadecimal.

- **End:** This part of the command always contains three characters. Two characters represent the checksum and the last character is a carriage return.

An example of a command which communicates with an OPTO-22 with address FF and writes to channel 1 is as follows:

Beginning End

>FF J 0002 000 78cr

Middle

In this example, >**FF** is the **beginning** part of the command, **J 0002 000** is the **middle** part of the command with **J** representing the **cmd**, **0002** the **position** and **000** the **data**. There is no **modifier** in the middle part of the command in this case. The **End** part of the command is **78cr**, where 78 is the checksum and cr is the carriage return character. A calculation of the checksum is presented in Appendix C. Further details on the OPTO-22 are available in the OPTO-22 users manual (1984).

Details about the development of the interface software are in a following section.

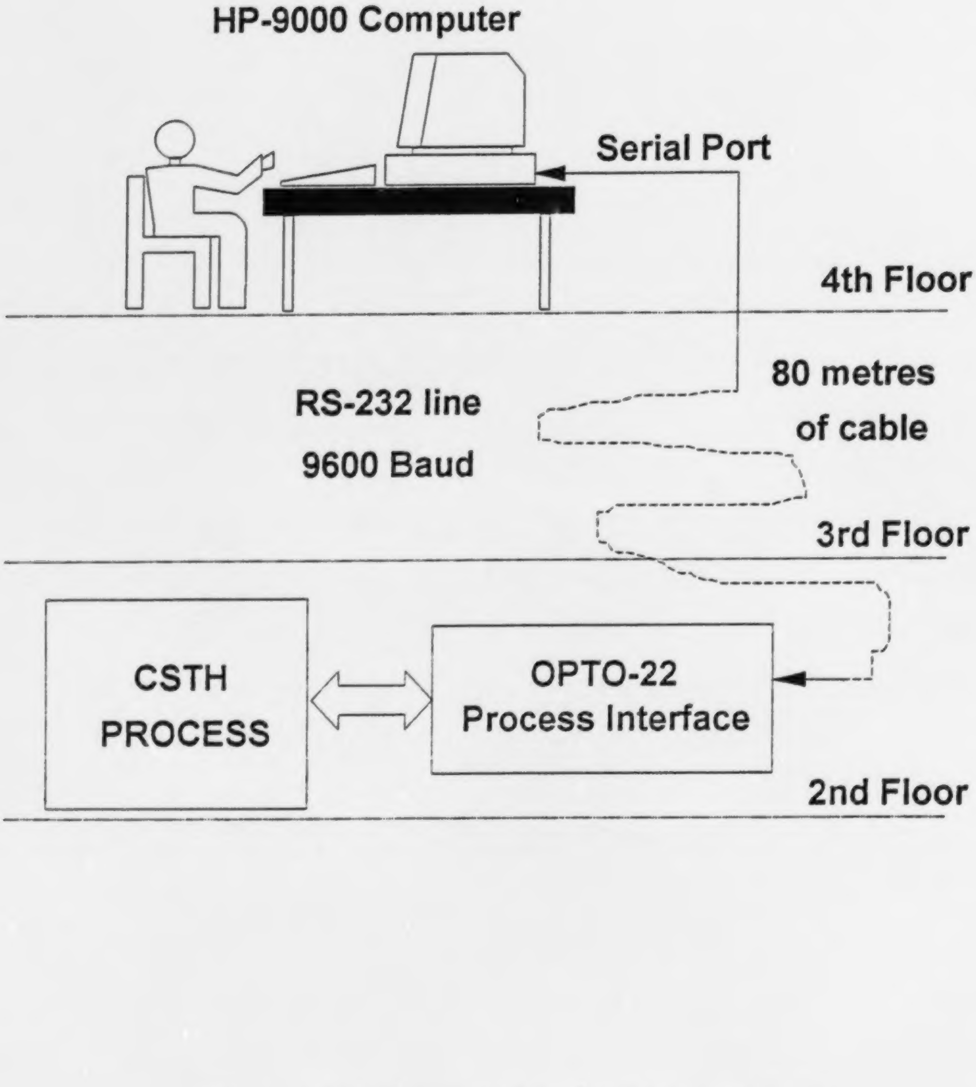


Figure 5.2 Hardware configuration of the equipment

5.2 Process model

Since GPC is a model based technique, it is important to get a good estimate of the model of the process. For the same set of tuning parameters, the better the model, the better the closed loop response will be. The model of the process is determined by providing separate steps in the cold water and the hot water flow rates and then recording the temperature and the level.

5.2.1 Step response model

The process model used is the one determined by Mutha (1990). Two models were determined, one for thermocouple #1 and the other for thermocouple #2. The model in the Laplace domain for thermocouple #1 is given by:

$$\begin{bmatrix} y1 \\ y2 \end{bmatrix} = \begin{bmatrix} \dfrac{-0.35e^{-4s}}{40s+1} & \dfrac{0.18e^{-4s}}{44s+1} \\ \dfrac{0.031}{s} & \dfrac{0.0223}{s} \end{bmatrix} \begin{bmatrix} u1 \\ u2 \end{bmatrix} \tag{5.1}$$

Notice the delay in this case is 4 seconds. The physical meaning of the inputs and outputs is:

- **y1**- The temperature of the water in the tank.
- **y2**- The level of the water in the tank.
- **u1**- The cold water valve position.
- **u2**- The hot water valve position.

A similar model was also estimated by Mutha for thermocouple #2. This model is as follows:

$$\begin{bmatrix} y1 \\ y2 \end{bmatrix} = \begin{bmatrix} \dfrac{-0.35e^{-32s}}{40s+1} & \dfrac{0.45e^{-28s}}{44s+1} \\ \dfrac{0.031}{s} & \dfrac{0.0223}{s} \end{bmatrix} \begin{bmatrix} u1 \\ u2 \end{bmatrix} \tag{5.2}$$

From these models it is apparent that a step change in u1 or u2 will result in an open loop unstable response, *i.e.* the water level in the tank will keep rising until it overflows. This type of open loop behaviour is referred to as an *integrator* type of response and is due to the 's' in the denominator of the transfer function models of y2. The physical nature of the level response to a step change in the cold and/or the hot water suggests that there should be some steady state value of the level since the level is self regulating *i.e.* the flow rate out from the tank increases as the level of water becomes higher. This, however is still true, but the time constant of the process is extremely large, and the observed *integrator* type of response is an approximation of what we see over a relatively short period of time.

In the experimental work done by Mutha (1990), GPC with constraints and fixed parameters was used to control the process given by equations 5.1 and 5.2. Mutha (1990) observed that the closed loop system becomes unstable for certain values of the output weighting. In order to validate this observation, the GPC algorithm was executed on the CSTH process. Figure 5.3 shows the results from this experiment. From this figure, we observe that for the output weighting γ_2=0.5 the closed loop system becomes unstable. The value of γ_1 is kept constant at 1.0. Looking carefully at the control signal at sample time 460 in figure 5.3, we observe that it is the control action (Hot and Cold water valves) which show a sudden movement when the output weighting γ_2, is set to 0.5. Thus the cause of the instability is the calculation of a strong control action by GPC. Since this is unacceptable in a situation where the output weighting is used to tune the closed loop response, something had to be changed (in the model or the tuning parameters) in order to make the process closed loop stable, irrespective of the value of the output weighting.

After some trial and error, the author found that if the models for the level response shown above in equations 5.1 and 5.2 were changed such that they did not contain the *integrator* type of model, but instead approximated the ramp type response by large time constants, then the closed loop system became stable irrespective of the value of the output weighting. These models were estimated by simulating the models for y2 and then fitting a model of the form

$$y = \frac{k}{\tau s + 1} \tag{5.3}$$

to them. The resulting models estimated for the level loop, for thermocouples #1 and #2 respectively, are as follows:

$$\begin{bmatrix} y1 \\ y2 \end{bmatrix} = \begin{bmatrix} \frac{-0.35e^{-4s}}{40s+1} & \frac{0.18e^{-4s}}{44s+1} \\ \frac{660}{5950s+1} & \frac{500}{5950s+1} \end{bmatrix} \begin{bmatrix} u1 \\ u2 \end{bmatrix} \tag{5.4}$$

and

$$\begin{bmatrix} y1 \\ y2 \end{bmatrix} = \begin{bmatrix} \frac{-0.35e^{-32s}}{40s+1} & \frac{0.18e^{-28s}}{44s+1} \\ \frac{660}{5950s+1} & \frac{500}{5950s+1} \end{bmatrix} \begin{bmatrix} u1 \\ u2 \end{bmatrix} \tag{5.5}$$

The models represented by equations 5.4 and 5.5 are used in this work. Two important things to note about these models is the large time constant and the high gain.

5.3 Process interface software development

A significant part of the experimental work was to connect the computer with the real process. Since the HP 9000 has a serial port, it is possible to use a RS-232 cable and connect it to the OPTO-22. In order to communicate with the OPTO-22, some software was written in 'C'. The purpose of this software is to convert the data put out by the OPTO-22 to numbers and to convert the numbers calculated by the GPC program to commands understandable by the OPTO-22. There were a number of steps in the process of developing, debugging and testing the software. Some of the steps are presented in the following sections.

5.3.1 Communication protocol

Communication protocol is the 'set of rules' which govern the method in which two pieces

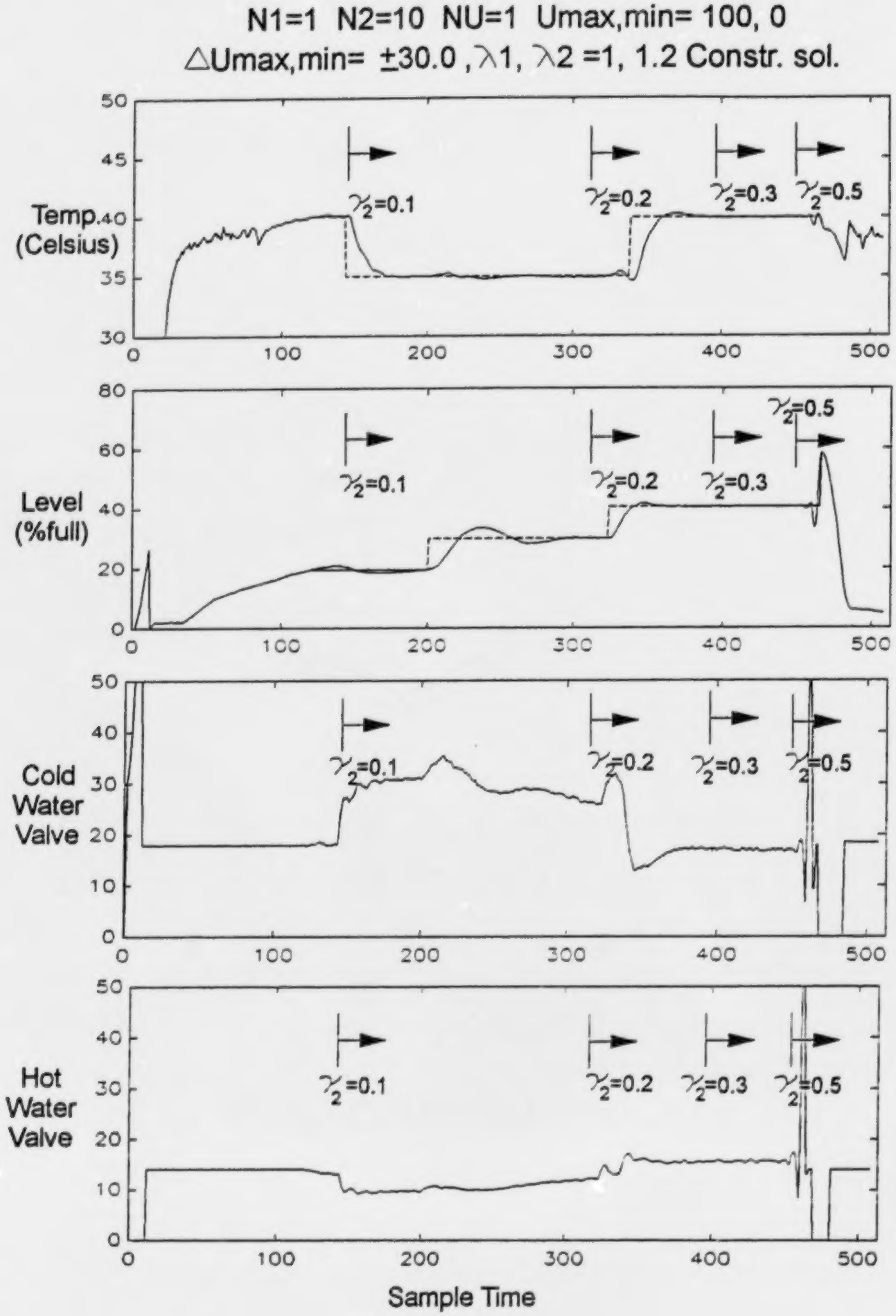


Figure 5.3 Effect of output weighting on closed loop response

of equipment 'talk' or communicate with each other. Since the objective of the software is to be able to convert a command from the OPTO-22 to numbers recognizable by a computer and vice versa, a common communication protocol becomes the first step in the software development process. The serial port on the HP 9000 computer is configured to the settings described in section 5.1.2. These settings are the same on the OPTO-22, ensuring proper communication. In addition to the settings of the baud rate and characteristics of the data word, an additional setting on the HP 9000 which defines the minimum number of characters read into a buffer, had to be changed from 4 to 1. This ensures that single character responses from the OPTO-22 can be read by the computer immediately, rather than wait until 4 characters are available to be read. This is very important from the point of view of error checking because most of the errors reported by the OPTO-22 are single character responses, and it is important to read these immediately, since this is a real-time, on-line application.

In order to check if the communication protocol was functioning correctly, a testing system was devised. The testing set up consisted of a HP 2392A monitor with a keyboard connected to the configured serial port of the HP 9000. The HP 2392A monitor was configured to the same communication protocol as the HP 9000. A small program was written in 'C' which wrote some characters to the HP 2392A monitor and then waited for a response. Typing some characters on the HP 2392A would then show up on the HP 9000 monitor. This completed the test and demonstrated that the serial port was functional and that the communication protocol was adequate.

5.3.2 Data integrity and program integration

The next stage in the software development was to connect the HP 9000 computer to the OPTO-22 two floors below. The cable was laid out and some tests were devised to ensure that data being read from and written to the process was correct. Another short program was written in 'C' which initialised the OPTO-22, wrote to certain channels and checked for error codes. The program also read the level and the temperature of the water in the tank. After extensive testing, the connection between the OPTO-22 and the HP 9000 was considered established and secure.

Once the hardware configuration was established, what was left to do was to integrate the program into the rest of the GPC code and have it execute every sampling instance, which was 2 seconds. Figure 5.4 presents the data transmission and program configuration aspect of the system. The process interface software is developed as a set of procedures which set data to and get data from the process. From figure 5.4 a hierarchy is clearly visible. The Process Supervisor is the software developed in G2. The GSI (G2 Standard Interface) software consists of a set of procedures written in the 'C' programming language. The purpose of the GSI software is to provide a one to one correspondence between the variables in G2 and the variables in the GPC program. This software also handles situations such as a pause in the execution of the knowledge base, resumption of execution and resetting the knowledge base. The GPC program is the heart of the control calculation. A number of procedures get called from this program, some of them being the procedures developed for the 'process interface software'.

5.4 Experimental results

A few experiments were performed to evaluate the GPC controller and the autotuning procedure. The initial tuning parameters were decided by using the set of rules developed in section 4.2.1 and the experience of Mutha(1990). The initial tuning parameters are as follows:

$N1 = 1$

$N2 = 10$

$NU = 1$

$\gamma_1 = 1.0$, $\gamma_2 = 0.2$

$\lambda_1 = 1.0$, $\lambda_2 = 1.2$

$\Delta U_{max,min} = \pm 10.0$

$U_{max,min} = 100, 0$

T-filter = $1-0.8q^{-1}$ (1st order T-filter)

Other important parameters are the inlet water temperatures. The Cold water temperature varies between 10 °C and 20 °C while the hot water temperature varies between 40 °C and

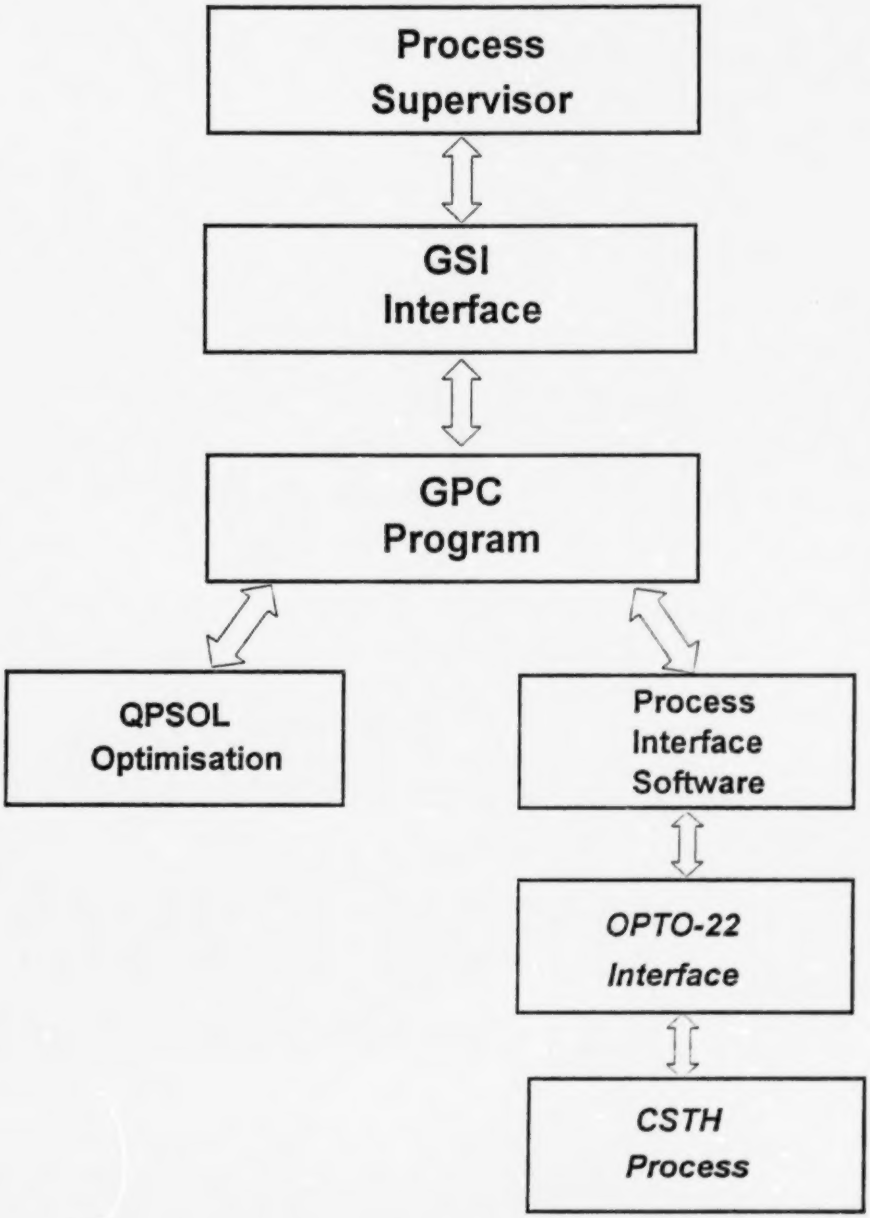


Figure 5.4 Data transmission and program configuration

54 °C. The pressure also varies in the cold and hot water lines due to intermittent water consumption in various parts of the building. The sampling time used for all experiments is 2 seconds. This sampling time is chosen keeping in mind the following experimental considerations:

- The water level in the tank rises very quickly, thus the sampling time should be fast, to detect the rise in level fairly quickly.
- The GPC and QPSOL programs being executed on the HP 9000 computer require sufficient time for execution. Since the QPSOL algorithm is a solution searching algorithm, the execution time increases if the solution search fails.

If a 1 second sampling interval is chosen then there is the possibility of insufficient execution time for the QPSOL algorithm on the odd occasion when the solution search fails. The next higher integer value of 2 seconds, is thus a more appropriate choice.

A number of experiments were performed. In all the experiments, the process is started up in simulation mode and then switched to manual, on-line, real-time operation mode. It is due to this switch that the reader might notice a sudden jump in process variables and control valve positions at the beginning of all the figures with experimental results. The manual mode configuration allows the user to set the cold and hot water valve positions to desired values. Once the process is at steady state, the process is switched to automatic mode. Details of the knowledge base are in section 2.4.4.

First an experiment is performed to establish a base case for comparison purposes. Subsequent experiments are performed to test various autotuning configurations. In all the experiments, the same set point trajectory is used so that results can be compared.

5.4.1 Results with autotuner disabled

A base case experiment is performed in order to compare results. The base case scenario has the tuning parameters set to the values mentioned in section 5.4 above. The result of this experiment is presented in figure 5.5. A few things to note about this experimental result are as follows:

- There is an overshoot of about 10% in the level when a set point change is made for the temperature loop at sample time 100.

- After the set point change is made at sample time 300, the temperature and level show some oscillations and there is an overshoot of about 10% in the level.
- After a set point change is made at sample time 380, the level seems to display growing oscillations.
- The final set point change at sample time 480 results in both the temperature and the level achieving steady states.

These results show that there is room for improvement in the response of the temperature and the level.

5.4.2 Results with autotuner enabled

Three different experiments were performed with the autotuner enabled. One experiment was performed to evaluate the performance of the regulatory supervisor by itself, another to evaluate the effect of the servo supervisor by itself and the final experiment was performed to evaluate the effect of the regulatory and the servo supervisor together. Details of the regulatory and the servo supervisor are available in section 2.4.4.

5.4.2.1 Results with regulatory supervision

The results with regulatory supervision are presented in figure 5.6. To refresh the reader's memory, the regulatory supervisor updates the T-filter and the output weighting automatically. The supervisor is activated at sample time 190. As seen from figure 5.6, the output weighting which has a default starting value of 1.0 and 0.2 for the temperature and the level loops respectively, is immediately updated. Since the set point trajectory is the same for all experiments, it is easy to compare the performance of the closed loop response. The following observations can be made from figure 5.6:

- From figure 5.6, when the first set point change is made at sample time 200, there is very little interaction in the level loop as compared to the set point change made at time 100 in figure 5.5 when there is an overshoot of about 10% in the level loop. Another example of the reduction in interaction can be seen at sample time 310 in figure 5.5 and sample time 415 in figure 5.6. The overshoot in the level loop in figure 5.5 at time 310 is about 8% while the overshoot in figure 5.6

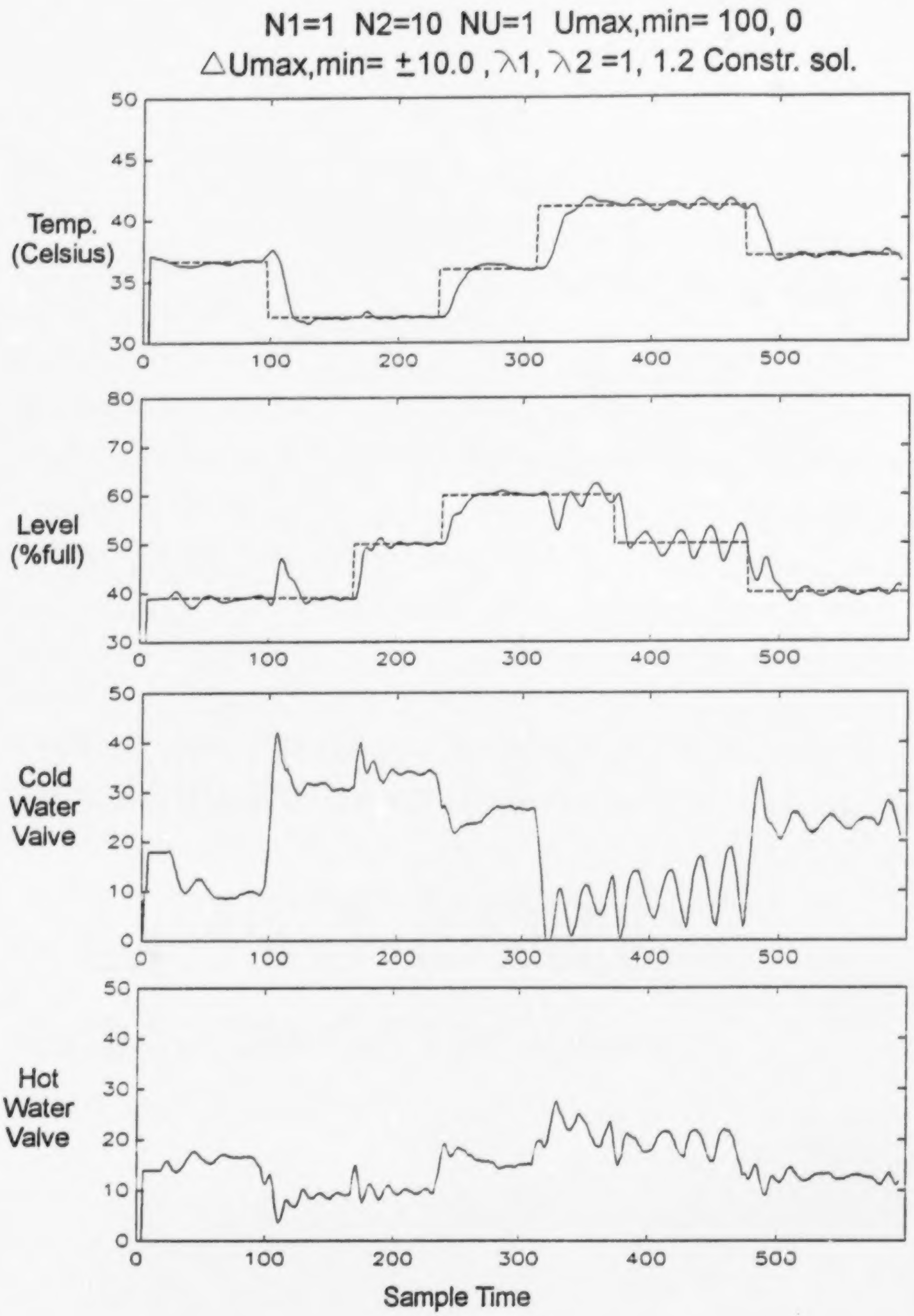


Figure 5.5 Experimental evaluation of GPC with autotuner disabled

at time 415 at the same operating conditions is less than 2%, indicating that tuning output weighting helps in reducing interactions.

- Besides the affect of decoupling the outputs, autotuning the output weighting and the T-filter can also help in stabilising a potentially unstable process. An example of this is in figure 5.5, between the sample time 380 and 470. Looking at this interval of time, there are growing oscillations in the level loop until a set point change is made at time 470, when the oscillations are controlled. A comparison of the same operating condition with tuned output weighting and T-filter in figure 5.6, between the sample time 460 to 510 shows absolutely no oscillations. This again shows the effectiveness of this tuning strategy and its use in real processes.

5.4.2.2 Results with servo supervision

Servo supervision involves the updating of the higher and lower limits on the outputs and tuning the value of N2 based on the rise time. Since from figure 5.6, we see that the final T-filter implemented is a high 1st order filter ($1-0.92q^{-1}$), a low second order filter is used in this experiment. Results are presented in figure 5.7. A few observations are as follows:

- Due to the presence of a second order T-filter, the trajectory of the control action is a lot smoother. However, since the output weighting is fixed, decoupling is not as good as when regulatory supervision was active. The set point change made at time sample time 190 results in overshoot of about 5% in the level loop. This is not as high as the 10% overshoot in figure 5.5 at sample time 100. The outputs stay within the output regulation band for all the set point changes.
- The value of N2 is set to 8 after sample time 200. This is done based on the rise time calculated for the level and the temperature. The impact of this tuning is not very profound due to the relatively small change in N2. Decreasing the value of N2 decreases the rise time for each of the outputs.

Overall, the use of the servo supervisor is to ensure that the outputs remain within a certain specified bound and attain the desired rise time via tuning N2. Both these criteria are met in this experiment.

5.4.2.3 Results with regulatory and servo supervision

Each tuning procedure does a specific task, be it tuning the T-filter or tuning the value

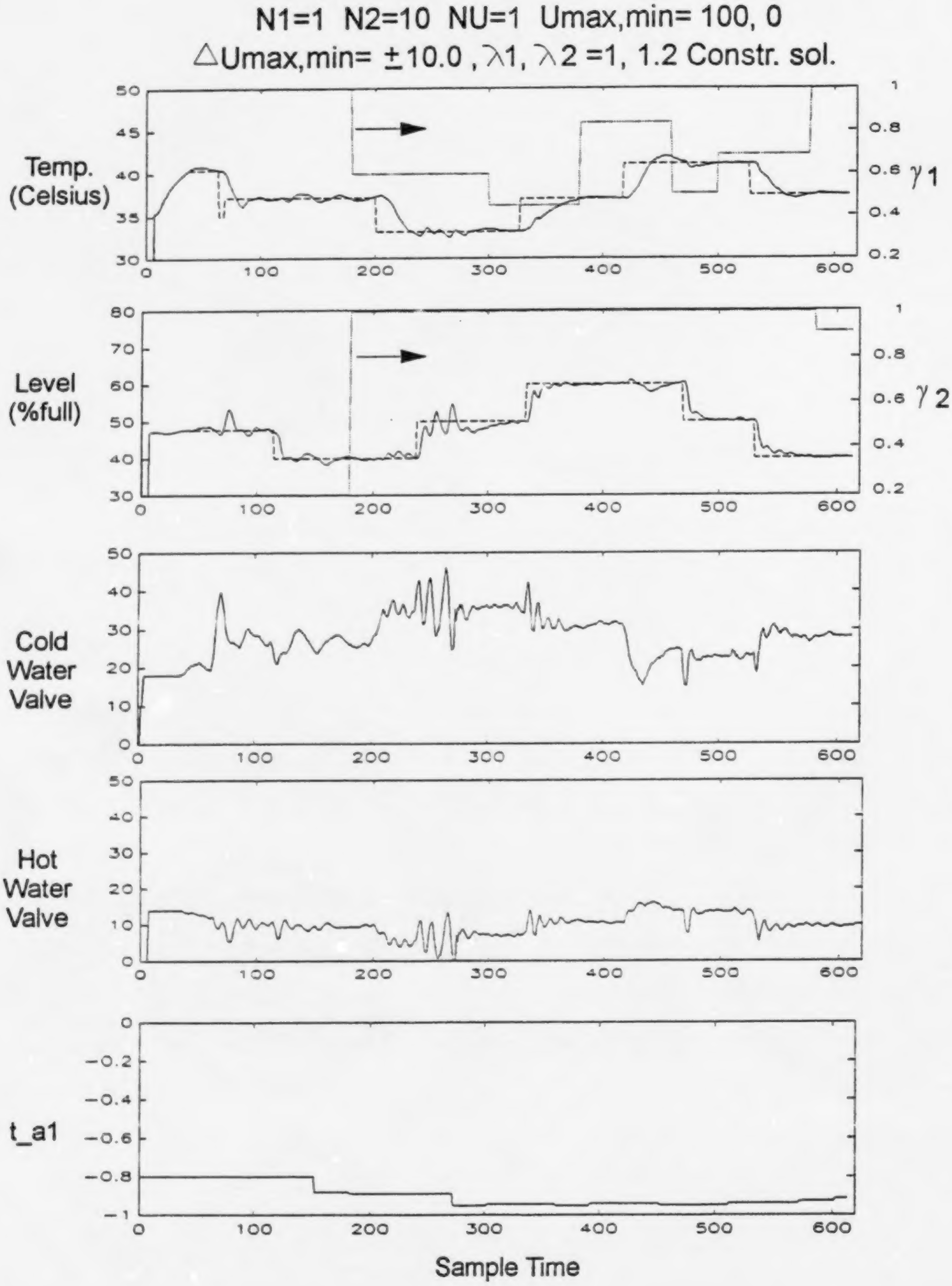

N1=1 N2=10 NU=1 Umax,min= 100, 0
$\triangle$Umax,min= ±10.0 , λ1, λ2 =1, 1.2 Constr. sol.

Figure 5.6 Experimental evaluation of regulatory supervision

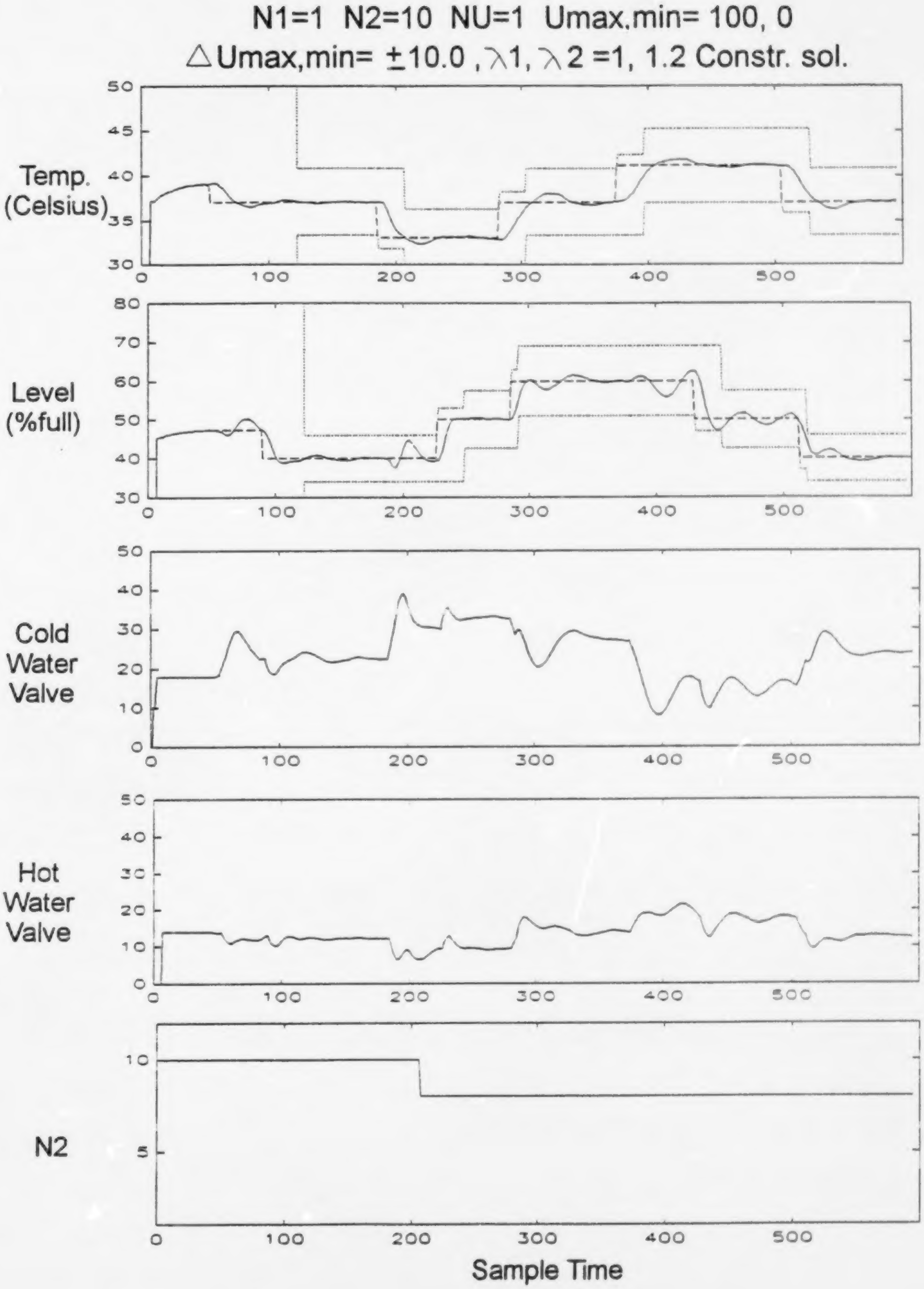


Figure 5.7 Experimental evaluation of servo supervision

of N2. However, the objective is to be able to keep all the process response criteria such as rise time within specification and at the same time decouple the temperature and level loops as much as possible. In order to take care of all these criteria, the regulatory and the servo supervisors have to be combined and implemented on the CSTH equipment. Results from this experiment are in figures 5.8 and 5.9. The regulatory supervisor is turned on at sample time 198 and the servo supervisor is turned on at sample time 240. Some observations are as follows:

- There is a sudden jump in the value of the T-filter when the regulatory supervision option is enabled at sample time 200. This jump results in an overshoot of about 6% in the level. This means that sudden changes in T-filter values are undesirable. The relaxation factor, w, described in section 2.4.4.2 can be decreased from 0.2 to a smaller value to make the T-filter tuning more gradual.
- The closed loop response of the level is faster than when autotuning was disabled (figure 5.5). However, the response of the temperature is quite slow. Due to this slow response, the value of N2 gets set to 5 from the initial value of 10. This too, does not help in decreasing the rise time of the temperature loop. The level loop however, becomes 'tightly tuned' as is evident from the oscillations at sample time 550 when a set point change is made in the level.Overall, the decoupling is good in this configuration, but there needs to be improvement in the speed of response of the temperature loop.

5.5 Conclusions

Due to the changing nature of chemical processes and the varied operating conditions, it is necessary to re-tune control loops on-line. Depending on the control strategy, the tuning method can be based on heuristics and knowledge of the operator or based on certain parameters which are readily available from the process. The tuning strategy implemented in this experimental evaluation of the Generalised Predictive Control algorithm is based on a mathematical formulation and certain rules of thumb.

The experiments performed clearly indicate that autotuning helps in

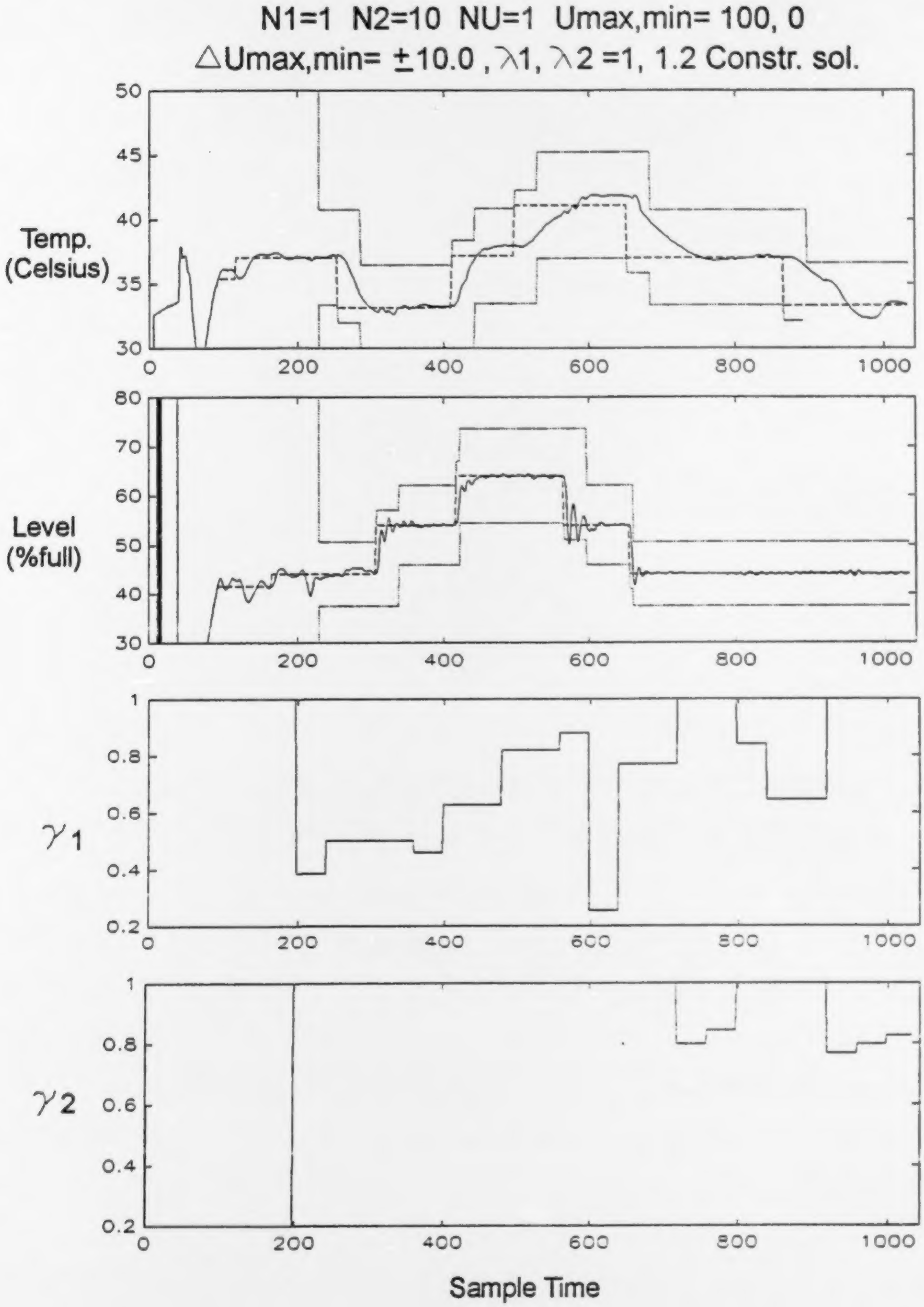
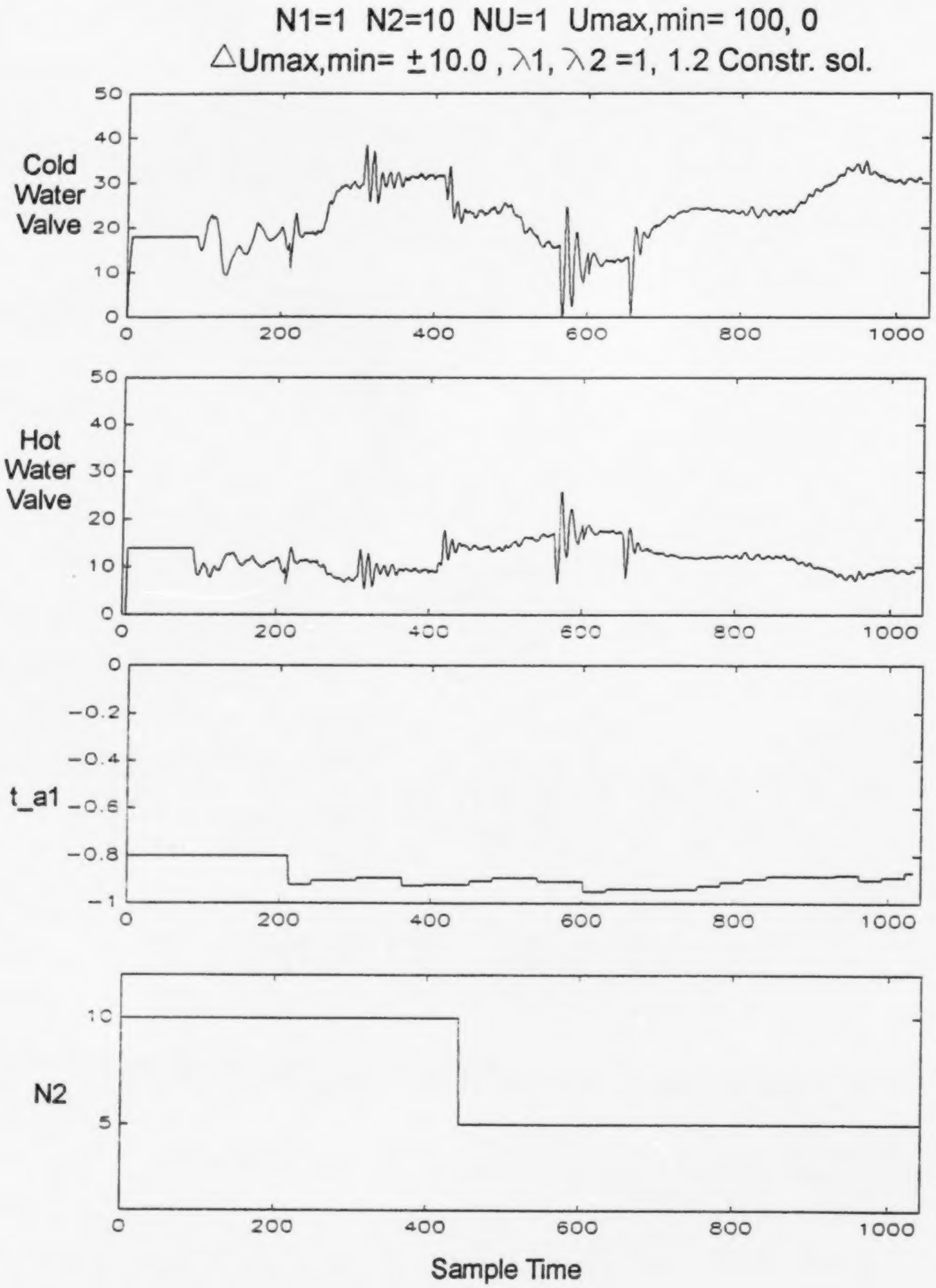


Figure 5.8 Results with regulatory and servo supervision enabled



N1=1 N2=10 NU=1 Umax,min= 100, 0
ΔUmax,min= ±10.0 , λ1, λ2 =1, 1.2 Constr. sol.

Figure 5.9 Results with regulatory and servo supervision enabled

improving the response of the process and decreases interactions among channels. The regulatory response supervisor is especially powerful in decoupling channels and choosing the 'optimum' amount of filtering. The results obtained after activating the regulatory response supervisor are much better than those obtained with no autotuning. The servo response supervisor helps in keeping the rise time within specified desired values and also allows a regulation band within which the outputs are required to remain. Since all these are user specified options, there is a lot of control in the hands of the user.

The tuning method implemented can be used in either an autotuning mode for processes which have a lot of changes in their operating points and operating conditions, or in deciding the tuning parameters for a certain set of operating conditions and operating points. The latter configuration is useful when computing time usage becomes important for large multivariable systems. Another important feature of this tuning strategy is that it is generic and can easily be applied to another plant, with equal or lesser number of inputs and outputs.

The importance of tuning can also be shown in cases when the closed loop system becomes unstable but can be stabilized with a proper set of tuning parameters. For an on-line implementation, time is very important in situations in which the plant seems to be going unstable. This tuning strategy has the ability to stabilize an unstable plant by automatically implementing the proper tuning parameters for an interactive multivariable system.

Chapter 6

Conclusions

For many years personal knowledge and rules of thumb have been passed down by word of mouth. This knowledge can be as subtle as a special ingredient in a cooking recipe or as sophisticated as a special procedure to be followed when a plant operator thinks that one of the process outputs is drifting away from the desired value. During the course of this transfer of knowledge from one person to the other, some details may be lost and some facts may be distorted. This is where expert systems help by not only capturing the knowledge of an expert in a specific domain, but also providing a means by which the knowledge of other experts can be added to an existing expert system knowledge base.

The purpose of this thesis is to extend and apply one such expert system called ALPS (Walther, 1992) to a relatively large, industrial size process and to evaluate the performance of this expert system experimentally. The knowledge base of ALPS contains rules which tune selected GPC parameters so that the process outputs have good tracking properties.

In order to put the need for tuning GPC into perspective, consider a process with n outputs and m inputs. With an increase in the number of outputs and inputs, the number of interactions affecting each output increase dramatically, making it harder to control the process and thus requiring better tuning of the GPC algorithm. The GPC algorithm which takes care of all the control calculations is applicable to processes with n outputs and m inputs.

The original tuning rules of ALPS designed for a 2-input 2-output process are applied to a larger 5-input 5-output problem, demonstrating the generic nature of the rules. However, some of the rules which provide procedural type functions such as rise time calculations, are modified because the original implementation method devised by Walther uses an exhaustive 'search method' to determine the status of outputs. Only 5 rules for the 5-input 5-output system are required after this modification, but if the exhaustive method of Walther were used, then 120 rules would be required.

Besides the domain specific problem solving nature of expert systems, the important concept of continuity and longevity of a knowledge base is demonstrated by the ease with which ALPS can be modified and enhanced for future applications.

A list of the contributions of this thesis are as follows:

1. Experimental evaluation of ALPS

The primary reason for limited on-line applications of expert systems is the lack of confidence and proof that an expert system will actually do a given task in a given time frame. The experimental evaluation of ALPS in this work is a step towards understanding the on-line requirements and performance of an expert system.

In order to implement ALPS on-line, process interface software was developed, the communication protocol established with the OPTO-22 process interface and the hardware link was tested for data integrity.

Experiments indicate that ALPS does a good job in tuning selected GPC parameters when poor initial estimates are provided. The resulting tuned GPC algorithm does a better job in controlling the process. Also, if an incorrect choice of initial tuning parameters causes the process to go unstable, then activating this tuning strategy can stabilize the process. It has also been demonstrated experimentally, that choosing a stable model for a process which is open loop unstable (an *integrator* type process), results in a stable closed loop system, no matter what the value of the output weighting. This is an improvement on the experiments done by Mutha (1990), in which choosing a *integrator* model for an *integrator* type process resulted in a stable closed loop system for certain values of output weighting only.

The applicability of a tuning strategy such as ALPS in an on-line industrial scenario is obvious because of the large number of control loops present in industry and the lack of a systematic and consistent manual method of tuning these loops.

2. Enhancing ALPS for a larger application

In process control, the complexity of problems grows exponentially as the dimensions of the problem are increased. For example, tuning a control algorithm for a 2-input, 2-output system is a lot easier than tuning a control algorithm for a 5-input 5-output problem. However, by using an expert systems approach, a generic tuning algorithm can be developed which can (in principle) handle a problem of any complexity. This has been demonstrated in this thesis by applying the existing ALPS knowledge base to a 5-input 5-output problem.

Although the tuning rules are generic and applicable to any plant, other rules such as the rules used to calculate the rise time of an output had to be modified to handle the larger application. The exhaustive approach used by Walther (1992) in developing ALPS for a 2-input 2-output system would result in a large number of rules for the larger 5-input 5-output system (${}^5P_5 = 120$ rules), which is not practical from an implementation point of view. In order to avoid this problem, a non-exhaustive approach is taken to write the rules, by using a few procedures which simplify the logic.

Additions were made to the graphical and numeric displays plus the user entry panels to accommodate the larger dimension problem. A manual control panel was also added so that the process could be simulated in manual control mode as well as automatic mode. The G2 Standard Interface or GSI program was modified as well for the larger application. The GSI program provides a one to one link between the variables of G2 and the variables of the external GPC program.

During the modification and testing procedure it was discovered that the problem being simulated was very sensitive to the precision of the variables used, thus all the programs were changed to double precision variables. This problem of numerical stability could be problem specific, but it is recommended that all programming for larger dimension problems should be done in double precision so that the closed loop characteristics of the process are not confused with the numerical stability of the algorithm. The change to double precision variables did not significantly affect the execution time of the program.

With the knowledge base in its present form, simulations can be performed for industrial systems with constraints, having up to 5 inputs and 5 outputs, square or non-square. ALPS determines the GPC tuning parameters required for a specific operating point. Thus ALPS can be used as a valuable off-line tool for advising the plant operator of the tuning parameters to be implemented.

3. Developing guidelines for choosing initial tuning parameters of MIMO GPC

Guidelines for choosing GPC parameters for a single-input single-output (SISO) system are available in Clarke *et al.* (1987) and a number of tuning strategies for SISO GPC are available in the paper by McIntosh *et al.* (1991). Some guidelines for tuning MIMO GPC are available in Mutha (1990) and Walther (1992), however, in this work, a guideline is developed for choosing *initial* GPC tuning parameters of a MIMO system. The choice of initial tuning parameters of GPC is very important because an incorrect choice can result in an unstable closed loop system.This is especially true for large MIMO systems. Also, quite often, some of the initial tuning parameters end up being the final tuning parameters, so their choice is important.

The guidelines developed in this thesis are based on observations made from a number of simulations of the 5-input 5-output heavy oil fractionater. For example, in the presence of model plant mismatch, choosing input or λ weighting based on the *sensitivity* of outputs to inputs can help reduce large variances in the outputs. Since the heavy oil fractionater problem is representative of a number of complex processes in industry, the guideline can be considered to be applicable to at least this class of processes.

4. Evaluating the applicability of expert systems to overspecified non-square systems

There are a number of processes which are non-square, *i.e.* the number of inputs to these processes is not equal to the number of outputs. If the number of inputs is less than the number of outputs, then the system cannot be controlled. However, when the number of inputs is greater than the number of outputs (overspecified case), then it has been shown by simulations that the steady state solution of the constrained GPC problem is non-unique *e.g.* the final steady state values of the input variables can be different for the same operating point. From a practical perspective, this means that if there is a cost associated with each of the inputs, then the total cost of inputs can be 'optimized' by judicious choice of inputs, using the extra degree(s) of freedom.

Rules which use extra degree(s) of freedom were added to the existing ALPS knowledge base. The rules are generic, yet easily modifiable for any specific scenario. A simulation with a 5-input 4-output process showed that all outputs still converge to their set points after the extra input is activated when one of the inputs saturates at its upper constraint. The extra input is considered expensive and is used only when required.

6.1 Suggestions for future work

1. Since ALPS has proven to be a useful tool for tuning the GPC algorithm, a similar tuning methodology could be developed for the DMC algorithm, which is widely used in industry. Such an expert system can be used for decision support initially, and eventually used on-line.

2. There is a lack of firm theoretical basis for tuning multivariable GPC. Most of the results on tuning multivariable GPC are derived from results on single-input, single-output GPC. Investigation of the closed loop theory of multivariable GPC can produce some insight into the tuning method which, in turn, can enhance the ALPS expert system knowledge base.

3. The idea of judicious use of extra degrees of freedom needs to be investigated further as well. If costs of inputs are important, then a suitable strategy should be

devised which guarantees minimum cost of inputs, while guaranteeing offset-free control.

4. The idea of using a stable model for an open loop unstable MIMO process can b investigated further. Experiments show that using a stable model for an *integrator* type process results in a stable closed loop system, no matter what the value of the output weighting. Experiments by Mutha (1990) on the same experimental equipment show that using an *integrator* model for the *integrator* process results in restrictions on the choice of the output weighting. The affect of output weighting on closed loop poles of a MIMO system with an *integrator* model can be studied further to theoretically confirm this experimental observation.

5. The ALPS tuning strategy can be improved by making it more robust to on-line changes of variables such as the T-filter. The calculation of the *rise time* can also be made more flexible so that it can be calculated correctly for processes which display an underdamped or a non-minimum phase step response. The present application of the *rise time* is applicable to overdamped processes only.

References

D.Y.Abramovitch, R.L.Kosut and G.F.Franklin, 1986. "Adaptive Control With Saturating Inputs", *Proc. 25th IEEE Conference on Decision and Control,* Athens, Greece, 848-852.

J.L.Alty and G.Johannsen, 1989. "Knowledge-based Dialogue for Dynamic Systems", *Automatica*, **25**, (6), 829-840.

G.R.Arulalan and P.B.Deshpande, 1986. "Synthesizing a Multivariable Predictive Algorithm for Noninteracting Control", *Proc. ACC,* Seattle, USA, 1378-1386.

K.E.Årzén, 1989. "An Architecture for Expert System Based Feedback Control", Automatica, **25**, (6), 813-827.

M.Aynsley, D.Peel and A.J.Morris, 1989. "A Real-Time Knowledge-Based System for Fermentation Control", *Proc. ACC,* Pittsburgh, **3**, 2239-2244.

J.G.Balchen, 1990. "The stability of 2x2 Multivariable Control Systems", *Modeling, Identification and Control,* **11**, (2), 97-108.

S.Bittanti and R.Scattolini, 1985. "Multivariable self-tuning control: A model following approach", *Int. J. Control,* **42**, (5), 1035-1047.

D.Bonvin and D.A.Mellichamp, 1987. "A Scaling Procedure for the Structural and Interaction Analysis of Dynamic Models", *AIChE Journal*, **33**, (2), 250-257.

U.Borison, 1979. "Self-Tuning Regulators for a class of Multivariable Systems", *Automatica*, **15**, 209-215.

J.M.Boyle, G.K.H.Pang and A.G.J.MacFarlane, 1989. "The development and implementation of MAID: a knowledge based support system for use in control system design",*Trans. Institute of Measurement and Control*, **11**, (1), 25-39.

D.M.Braunstein, A.S. Brown and D.L.Drew, 1989. "A Knowledge-Based System for Process Monitoring and Control in Propellant Manufacturing", *Proc. ISA*, Philadelphia, Pennsylvania, 457-464.

D.W.Clarke, C.Mohtadi and P.S.Tuffs, 1987. "Generalized Predictive Control-Part 1 The basic algorithm", *Automatica*, **23**, (2), 137-148.

D.W.Clarke, C.Mohtadi and P.S.Tuffs, 1987. "Generalized Predictive Control-Part 2 Extensions and interpretations", *Automatica*, **23**, (2), 149-160.

D.W.Clarke and C.Mohtadi, 1987. "Properties of Generalised Predictive Control", *Proc. 10th IFAC Triennial World Congress*, Munich, FRG, 65-76.

D.W.Clarke, 1991. "Adaptive Generalized Predictive Control", *Presented at the 4thInternational conference on Chemical Process Control*, South Padre Island, Texas.

C.R.Cutler and R.T.Perry, 1982. "Real Time Optimisation with Multivariable Control is required to maximize profits", *Computers and Chemical Engineering*, 7, (5), 663-667.

C.R.Cutler and B.L.Ramaker, 1979. "Dynamic Matrix Control - A Computer Control Algorithm", *Presented at the meeting of the AIChE*, Houston, Texas.

C.R.Cutler and R.B.Hawkins, 1987. "Constrained Multivariable Control of a Hydrocracker Reactor", *Proc. of ACC*, Minneapolis, Minnisota, 1014-1020.

C.R.Cutler and R.B.Hawkins, 1988. "Applications of a large Predictive Multivariable Controller to a Hydrocracker Second Stage Reactor", *Proc. of ACC*, Atlanta, Georgia, 284-291.

C.R.Cutler and S.G.Finlayson, 1988. "Multivariable Control of a C3C4 Splitter Column", *Presented at the national meeting of AIChE*, New Orleans, Louisiana.

J.M.Dion, L.Dugard, A.Franco, N.M.Tri and D.Ray, 1991. "MIMO Adaptive Constrained Predictive Control Case Study: An Environmental Test Chamber", *Automatica*, **2** (4), 611-626.

R.Doraiswami and J.Jiang, 1989. "Performance monitoring in Expert Control Systems", *Automatica*, **25**, (6), 799-811.

L.Dugard and J.M.Dion, 1985. "Direct Adaptive Control for Linear Multivariable Systems", *Int. J. Control*, **42**, (6), 1251-1281.

G.Favier, 1987. "Self-Tuning Long-Range Predictive Controllers", *10th IFAC Triennial World Congress*, Munich, FRG, 83-90.

P.E.Gill, W.Murray, M.A.Saunders and M.H.Wright, 1984. "Users Guide for QPSOL: A Fortran Package for Quadratic Programming", *Technical Report SOL 84-6*, Dept. of Operations Research, Stanford University, Stanford, U.S.A.

B.J.Glass, W.K.Erickson and K.J.Swanson, 1991. "TEXSYS: A Large Scale Demonstration of Model-Based Real-Time Control of A Space Station Subsystem", *Proc. 7th IEEE conference on AI Applications*, Miami Beach, Florida, 378-384.

P.J.Glova, D.L.Snowden and F.H.Yocum, 1989. "Multivariable Controllers Applied at Plastics Facility", *Presented at the meeting of the ISA*, Philadelphia, Pennsylvania.

M.Haest, G.Bastin, M.Gevers and V.Wertz, 1990. "ESPION: An Expert System for System Identification", *Automatica*, **26**, (1), 85-95.

D.A.Hokanson,B.G.Houk and C.R.Johnsston, 1989. "DMC Control of a Complex Refrigerated Fractionator", *Presented at the meeting of the ISA*, Philadelphia, Pennsylvania.

J.R.James, D.K.Frederick and J.H.Taylor, 1987. "Use of expert systems programming techniques for the design of lead-lag compensators", *Proc. IEE*, **134**, part D, (3), 137-144.

D.E.Lamb, P.Dhurjati, D.L.Chester and J.C.Hale, 1985. "An Academic / Industry Project to Develop an Expert System for Chemical Process Fault Detection", *Presented at the annual meeting of the AIChE*, Chicago.

G.G.Leininger, 1977. "Diagonal Dominance using Function Minimisation Algorithms", *Proc. 4th IFAC International Symposium*, Fredrickton, Canada, 105-112.

W.E.Leithead and J.O'Reilly, 1991. "Performance issues in the individual channel design of 2-input, 2-output systems, part 1: Structural Issues", *Int. J. Control*, **54**, (1), 47-82.

D.R.Lewin and M.Morari, 1988. "ROBEX: An expert system for Robust control synthesis", *Computers and Chemical Engineering*, **12**, (12), 1187-1198.

A.R.McIntosh, S.L.Shah and D.G.Fisher, 1991. "Analysis and tuning of Adaptive Generalised Predictive Control", *Can. J. of Chem. Eng.*, **69**, (2), 97-110.

A.I.Mees and P.E.Rapp, 1977. "Stability Criteria for Multiple Loop Nonlinear Feedback Systems", *Proc. 4th IFAC International Symposium*, Fredrickton, Canada, 183-188.

M.Minasi, 1990. "Bayes and Simple Expert Systems", *AI Expert,* August, 13-15.

C.Mohtadi, S.L.Shah and D.G.Fisher, 1991. "Frequency response characteristics of MIMO GPC", *Proc. European Control Conference,* Grenoble, France, **2**, 1845-1850.

R.L.Moore, L.Hawkinson, M.Levin, A.Hofmann, M.David, B.Matthews and J.Allard, 1988. "The G2 Real-Time Expert System", *Proc. ISA, International Conference and Exhibit,* **43**, (4), 1625-1633.

R.L.Moore, L.B.Hawkinson, C.G.Knickerbocker and L.M.Churchman, 1984. "A Real-Time Expert System for Process Control", *Proc. 1st IEEE Conference on ArtificialIntelligence Applications,* Denver, Colorado, 569-576.

R.L.Moore and M.A.Kramer, 1986. "Expert Systems in On-Line Process Control", *Proc. 3rd International Conference on Chemical Process Control,* Asilomar, California, 839-867.

R.K.Mutha, 1990. *Constrained Long Range Predictive Control*, M.Sc. Thesis, Dept. of Chem. Eng., University of Alberta.

M.A.Nevins, 1989. "The building of a Nuclear Power Knowledge-Based Control System", *Proc. ACC,* Pittsburgh, **3**, 2230-2232.

K.Noderer and H.Henein, 1990. "A Survey on the use of Expert Systems in the Iron and Steel Industry", *Presented at the Ironmaking Conference, Detroit. Iron Steel Society,* Warrendale, PA.

O.D.I.Nwokah and R.Perez, 1991. "On Multivariable Stability in Gain Space", Automatica, **27**, (6), 975-983.

F.Ohkawa and Y.Yonezawa, 1982. "A discrete model reference adaptive control system for a plant with input amplitude constraints", Int. J. Control, **36**, (5), 747-753.

J.O'Reilly and W.E.Leithead, 1991. "Multivariable Control by 'individual channel design'", *Int. J. Control,* **54**, (1), 1-46.

G.K.H.Pang, 1992. "A knowledge environment for an Interactive Control System Design Package, *Automatica* , **28**, (3), 473-491.

S.Park, 1988. "The Application of an Optimized DMC Multivariable Controller to the PCC catalytic cracking unit", *Presented at: International symposium Advanced Process Supervision and Real-Time Knowledge Based Control,* University of Newcastle Upon Tyne, U.K.

A.N.Payne, 1986. "Adaptive one-step-ahead control subject to an input-amplitude constraint", *Int. J. Control*, **43**, (4), 1257-1269.

Y.Peng and R.Hanus, 1991. "A Tuning Strategy for Generalized Predictive Control", *Control-Theory and Advanced Technology*, 7, (1), 153-166.

D.M.Prett and M.Morari, 1986. *Shell Process Control Workshop*, Butterworth publishers, Stoneham, MA.

M.Rao, J.J.-P.Tsai and T.S.Jiang, 1988. "An Intelligent Decisionmaker for OptimalControl", *Applied Artificial Intelligence*, **2**, 285-305.

D.Roberts, R.Herrod and L.Tietz, 1988. "On-Line Real-Time Expert System for Cooling Water", *Proc. 49th International Water Conference*, Pittsburgh, Pennsylvania, 123-136.

R.Scattolini and D.W.Clarke,1985. "Multivariable model-following self-tuning control with offset rejection", *Int. J. Control*, **42**, (6), 1309-1322.

S.L.Shah, C.Mohtadi and D.W.Clarke, 1987. "Multivariable adaptive control without a prior knowledge of the delay matrix", *Systems and Control Letters*, **9**, 295-306.

F.G.Shinskey, 1977. "The Stability of Interacting Control Loops with and without Decoupling", *Proc. 4th IFAC International Symposium*, Fredricton, Canada, 21-30.

C.D.Tebutt, 1992. "An Expert system for multivariable Controller design", *Automatica*, **28**, (3), 463-471.

M.Thomson, P.G.Cassidy and J.Sandoz, 1989. "Automatic tuning of PID controllers using a combined time-and frequency-domain method", *Trans. Institute of Measurement and Control*, **11**, (1), 40-47.

D.Tran and C.R.Cutler, 1989. "Dynamic Matrix Control on Benzene and Toluene Towers", *Presented at: The meeting of the ISA*, Philadelphia, Pennsylvania.

A.H.VanHoof, C.R.Cutler and S.G.Finlayson, 1989. "Applications of a Constrained Multivariable Controller to a Hydrogen Plant", *Presented at: ACC*, Pittsburgh, Pennsylvania.

R.Walther, 1992. *Expert Long Range Predictive Control*, M.Sc. Thesis, Dept. of Chem. Eng., University of Alberta.

W.-J.Wang,C.-C.Kao and C.-S.Chen, 1991. "Stabilisation, Estimation and Robustness for Large Scale Time-Delay Systems", *Control-Theory and Advanced Technology*, 7, (4), 569-585.

J.R.Winkelman, 1987. "A Tuning Algorithm for a Multivariable Controller", *IFAC 10th Triennial World Congress*, Munich, FRG, 185-190.

C.Zhang and R.J.Evans, 1987. "Amplitude constrained adaptive control", *Int. J. Control*, **46**, (1), 53-64.

Reference Manuals

Opto 22: Analog user's manual, 1st edition, 1984. 15461, Springdale Street, Huntington Beach, CA 92649.

G2 Reference manual, Version 2.0, Gensym Corp., Cambridge Park Drive, Cambridge, MA 02140.

Appendix A

This Appendix contains procedures used to calculate the rise time of an output for a 5-input 5-output process. Procedures used previously for a 2-input 2-output process are also listed for comparison purposes.

This procedure is for a 2-input 2-output process

```
            OBSERVE_2X2_PLANT, a procedure
Notes                            OK
User restrictions                none
Tracing and breakpoints          default
Default procedure priority       1
observe_2x2_plant(PLANT:class 2by2-plant)
OUTP:class mimo_output;
estimated_tr:quantity;
OUTP1:class mimo_output;
OUTP2:class mimo_output;
begin
allow other processing;
if( for every mimo_output OUTP connected to PLANT (
    the servo_status of OUTP = 0))
then return;

if( for every mimo_output OUTP connected to PLANT (
    the servo_status of OUTP - the value of the
       servo_status of OUTP  as of 2 seconds ago =
       0))
then return;

OUTP1 =(the mimo_output
       connected at the outlet-port-1 of PLANT);
OUTP2 =(the mimo_output
       connected at the outlet-port-2 of PLANT);

if( the servo_status of OUTP1 =0 and
    the servo_status of OUTP2 =1 and (
   (the servo_status of OUTP2 - the value of the
      servo_status of OUTP2 as of 2 seconds ago )
      =1))
then begin
      start find_tr(OUTP2,OUTP1);
      return;
      end;

if( the servo_status of OUTP1 =1 and
    the servo_status of OUTP2 =0 and (
   (the servo_status of OUTP1 - the value of the
      servo_status of OUTP1 as of 2 seconds ago )
      =1))
then begin
      start find_tr(OUTP1,OUTP2);
      return;
      end;
```

```
if ((the servo_status of OUTP1 - the value of the
   servo_status of OUTP1 as of 2 seconds ago ) =1)
then start servo_wait_cycle(OUTP1);

if ((the servo_status of OUTP2 - the value of the
   servo_status of OUTP2 as of 2 seconds ago ) =1)
then start servo_wait_cycle(OUTP2);

end
```

This procedure is for a 5-input 5-output process

```
          OBSERVE_5X5_PLANT, a procedure
Notes                             OK
User restrictions                 none
Tracing and breakpoints           default
Default procedure priority        1
observe_5x5_plant(PLANT:class 5by5_plant)
OUTP:class mimo_output;
estimated_tr:quantity;
OUTP1:class mimo_output;
OUTP2:class mimo_output; OUTP3:class mimo_output;
OUTP4:class mimo_output; OUTP5:class mimo_output;
begin
allow other processing;
if( for every mimo_output OUTP connected to PLANT (
    the servo_status of OUTP = 0))
then return;

if( for every mimo_output OUTP connected to PLANT (
    the servo_status of OUTP - the value of the
       servo_status of OUTP  as of 5 seconds ago =
       0))
then return;

OUTP1 =(the mimo_output
       connected at the outlet-port-1 of PLANT);
OUTP2 =(the mimo_output
       connected at the outlet-port-2 of PLANT);
OUTP3 =(the mimo_output
       connected at the outlet-port-3 of PLANT);
OUTP4 =(the mimo_output
       connected at the outlet-port-4 of PLANT);
          OUTP5 =(the mimo_output
       connected at the outlet-port-5 of PLANT);
if( the servo_status of OUTP1 =1  and the
   servo_status of OUTP2 =0 and  the servo_status
   of OUTP3 =0 and  the servo_status of OUTP4 =0
   and  the servo_status of OUTP5 =0 and
   risetimeprime = 1)
then begin
      start find_tr(OUTP1);
      return;
      end; if (the servo_status of OUTP1 >= 1)
then start servo_wait_cycle(OUTP1);

if( the servo_status of OUTP2=1 and the
   servo_status of OUTP1 =0 and  the servo_status
   of OUTP3 =0 and  the servo_status of OUTP4 =0
   and  the servo_status of OUTP5 =0 and
   risetimeprime = 1)
then begin
      start find_tr(OUTP2);                   return;
         end; if (the servo_status of OUTP2 >= 1)
then start servo_wait_cycle(OUTP2);
   if  ( the servo_status of OUTP3 = 1 and the
   servo_status of OUTP1 =0 and  the servo_status
   of OUTP2 =0 and  the servo_status of OUTP4 =0
   and  the servo_status of OUTP5 =0 and
   risetimeprime = 1)
then begin
      start find_tr(OUTP3);
      return;
      end; if (the servo_status of OUTP3 >= 1)
then start servo_wait_cycle(OUTP3);

if( the servo_status of OUTP4 =1 and the
   servo_status of OUTP1 =0 and  the servo_status
   of OUTP2 =0 and  the servo_status of OUTP3 =0
   and  the servo_status of OUTP5 =0 and
   risetimeprime = 1)
then begin
      start find_tr(OUTP4);
      return;
      end; if (the servo_status of OUTP4 >= 1)
then start servo_wait_cycle(OUTP4);
   if( the servo_status of OUTP5 =1 and the
   servo_status of OUTP1 =0 and  the servo_status
   of OUTP2 =0 and  the servo_status of OUTP3 =0
   and  the servo_status of OUTP4 =0 and
   risetimeprime = 1)
then begin
      start find_tr(OUTP5);
      return;
      end;
        if (the servo_status of OUTP5 >=1)
then start servo_wait_cycle(OUTP5);
end
```

This procedure is for a 2-input 2-output process

```
                FIND_TR, a procedure
Notes                          OK
User restrictions              none
Tracing and breakpoints        default
Default procedure priority     1
find_tr(OUTP1:class mimo_output,
         OUTP0:class mimo_output)
old_ysp,new_ysp:quantity;
watch_for_y,loop_counter,tr:quantity;
current_y,current_y_zm1:quantity;
current_y_zm2:quantity;
delta_ysp,slope:quantity;
begin
   allow other processing;
   inform the operator for the next 40 seconds
      that "
   find_tr:
   Entering with [the name of OUTP1]
   connected to [the name of the mimo_plant
   connected to OUTP1]";
repeat
   collect data
   new_ysp= the ysp of OUTP1;
   old_ysp =the value of the ysp of OUTP1
               as of 2 second ago;  end;
   delta_ysp=new_ysp - old_ysp;
   exit if(delta_ysp /= 0.0);
    allow other processing;
end;
   watch_for_y=old_ysp+0.8*delta_ysp;
collect data
current_y=the y of OUTP1;
current_y_zm1=the value of the y of OUTP1 as of 2
   seconds ago;
current_y_zm2=the value of the y of OUTP1 as of 4
   seconds ago;
end;
   for loop_counter=0 to 40 by 2
   do
     wait for 2 seconds;
     current_y_zm2=current_y_zm1;
     current_y_zm1=current_y;
     collect data
     current_y= the y of OUTP1;
     end;
     slope=(current_y - current_y_zm2)/4.0;
inform the operator for the next 6 seconds that "
```

```
loop_counter= [loop_counter]
current_y=[current_y]
current_y_zm1=[current_y_zm1]
current_y_zm2=[current_y_zm2]
watch_for_y=[watch_for_y]
slope=[slope]";
      exit if(( if (delta_ysp > 0.0) then
          (current_y > watch_for_y) else
           (current_y < watch_for_y)));
    end;
    tr=loop_counter + (watch_for_y  -
       current_y_zm1)/slope;
    wait for (40 - loop_counter);
   if(loop_counter >= 40) then
    begin
    inform the operator for the next 10 seconds
       that "
    find_tr:
    Exiting with [the name of OUTP1]
    connected to [the name of the mimo_plant
    connected to OUTP1]
    UNABLE to find tr, window expired";
    conclude that the servo_status of OUTP1 = (the
       servo_status of OUTP1 - 1);
   return;
   end;
    if(the servo_status of OUTP1 =1 and
       the servo_status of OUTP0=0) then
       conclude that the observed_tr of OUTP1 = tr;


    conclude that the servo_status of OUTP1 = (the
       servo_status of OUTP1 - 1);
    inform the operator for the next 10 seconds
       that "
    find_tr:
    Exiting with [the name of OUTP1]
    connected to [the name of the mimo_plant
    connected to OUTP1] ";
end
```

This procedure is for a 5-input 5-output process

```
               FIND_TR, a procedure
Notes                         OK
User restrictions             none
Tracing and breakpoints       default
Default procedure priority    1
find_tr(OUTP:class mimo_output)
old_ysp,new_ysp:quantity;
watch_for_y,loop_counter,tr:quantity;
current_y,current_y_zm1:quantity;
current_y_zm2:quantity;
delta_ysp,slope:quantity;
begin
   allow other processing;
   inform the operator for the next 20 seconds
      that "
   find_tr:
   Entering with [the name of OUTP]
   connected to [the name of the mimo_plant
   connected to OUTP]"; conclude that flag_tr = 1;
repeat
   collect data
   new_ysp= the ysp of OUTP;
   old_ysp =the value of the ysp of OUTP
                as of 5 second ago;  end;
   delta_ysp=new_ysp - old_ysp;
   exit if(delta_ysp /= 0.0);
    allow other processing;
end;
   watch_for_y=old_ysp+0.8*delta_ysp;
collect data
current_y=the y of OUTP;
current_y_zm1=the value of the y of OUTP as of 5
   seconds ago;
current_y_zm2=the value of the y of OUTP as of 10
   seconds ago;
end;
   for loop_counter=0 to 200 by 5
   do
     wait for 5 seconds;
     current_y_zm2=current_y_zm1;
     current_y_zm1=current_y;
     collect data
     current_y= the y of OUTP;
     end;
     slope=(current_y - current_y_zm2)/10.0;
inform the operator for the next 5 seconds that "
loop_counter= [loop_counter]
```

```
current_y=[current_y]
current_y_zm1=[current_y_zm1]
current_y_zm2=[current_y_zm2]
watch_for_y=[watch_for_y]
slope=[slope]";
     exit if(( if (delta_ysp > 0.0) then
        (current_y > watch_for_y) else
         (current_y < watch_for_y)));
   end;
   tr=loop_counter + (watch_for_y  -
      current_y_zm1)/slope;
  if(loop_counter >= 200) then
   begin
   inform the operator for the next 10 seconds
      that "
   find_tr:
   Exiting with [the name of OUTP]
   connected to [the name of the mimo_plant
   connected to OUTP]
   UNABLE to find tr, window expired"; conclude
      that flag_tr = 0;
   conclude that the servo_status of OUTP = (the
      servo_status of OUTP - 1);
  return;
  end;
   if(the servo_status of OUTP=1 ) then begin
      conclude that flag_tr = 0;
      conclude that the observed_tr of OUTP= tr;
         end;
  conclude that the servo_status of OUTP = (the
     servo_status of OUTP - 1);
   inform the operator for the next 10 seconds
      that "
   find_tr:
   Exiting with [the name of OUTP]
   connected to [the name of the mimo_plant
   connected to OUTP] ";
end
```

Appendix B

This Appendix contains a rule and a procedure added on to the ALPS knowledge base for input saturation detection of a 5-input 4-output process.

This rule calls the procedure named 'INPUT_SATURATION'

```
                                        INPUT_SAT_RULE, a rule
Options                                     not invocable via backward chaining, not invocable
                                               via forward chaining, may cause data seeking,
                                               may cause forward chaining
Notes                                       OK
User restrictions                           none
Names                                       INPUT_SAT_RULE
Tracing and breakpoints                     default
for any mimo_plant PLANT for any mimo_input INP                    connected to PLANT
   for any mimo_output OUTP     connected to PLANT          unconditionally start
   INPUT_SATURATION(PLANT)
Scan interval                               15 seconds
Focal classes                               none
Focal objects                               none
Categories                                  none
Rule priority                               6
Depth first backward chaining precedence    1
Timeout for rule completion                 use default
```

The INPUT_SATURATION procedure

```
          INPUT_SATURATION, a procedure
Notes                        OK
User restrictions            none
Tracing and breakpoints      default
Default procedure priority   1
input_saturation (PLANT: class mimo_plant)
max_u,u1,u2,u3,max_dev_y: quantity;
old_max_u,old_u1,old_u2,old_u3 : quantity;
   OUT: class mimo_output;
begin
   collect data
   u1 = the u of input1 ;
   u2 = the u of input2 ;
   u3 = the u of input3 ;
   old_u1 = the value of the u of input1 as of 10
   seconds ago;                            old_u2 =
   the value of the u of input2 as of 10 seconds
   ago ;                            old_u3 = the
   value of the u of input3 as of 10 seconds ago;
   end;                                     max_dev_y
   = (the maximum over each mimo_output OUT
   connected to PLANT of (the standard deviation
   of the y of OUT during the last 20 seconds));
   {inform the operator that " max_dev_y =
   [max_dev_y] ";}
    max_u = max(abs(u1),abs(u2),abs(u3));
    old_max_u =
       max(abs(old_u1),abs(old_u2),abs(old_u3));
       if(max_u >= 0.45 and abs(max_u - old_max_u)
       = 0.0) then do in parallel inform the
       operator for the next 5 seconds that
       "Saturated Input: Using the extra degree of
       freedom for controlling the process"; set
       the lambda of input5 to 0.01; end;
     if(max_dev_y <= 0.001) then do in parallel
       set the lambda of input5 to 10000.0;
       inform the operator for the next 5
       seconds that "Process at Steady State:
       freezing extra input to current value";
       end                             end
```

Appendix C

This Appendix contains an example calculation of the Checksum for the OPTO-22 process interface.

The checksum is calculated for the following Hexadecimal string:

FFJ0002000

The ASCII equivalent of the characters contained in the above string is as follows:

F = 70

J = 74

0 = 48

2 = 50

The sum of all the characters in the Hexadecimal string is as follows:

Sum = 2*70 + 74 + 6*48 + 50 = 632

Now divide 632 by 256 (FF in Hexadecimal):

632/256 = 2 + 120/256

The checksum is the Hexadecimal equivalent of the remainder (120) from the above division.

$120_{10} = 78_{16}$

Checksum = 78